T.C.
MİMAR SİNAN ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
İSTATİSTİK ANABİLİM DALI
YÜKSEK LİSANS TEZİ

VERİ MADENCİLİĞİNDE KULLANILAN TEKNİKLER VE BİR UYGULAMA

Elif Özge ÖZDAMAR
DANIŞMAN : Prof.Dr. M. Kemal YOĞURTÇUGİL

İSTANBUL – TEMMUZ 2002

Elif Özge Özdamar tarafından hazırlanan "Veri Madenciliğinde Kullanılan Teknikler ve Bir Uygulama" adlı araştırmanın Yüksek Lisans Tezi olarak uygun olduğunu onaylarım.

İmza

Prof.Dr. M. Kemal YOĞURTÇUGİL

Bu çalışma Mimar Sinan Üniversitesi Fen Bilimleri Enstitüsü İstatistik Anabilim Dalında Yüksek Lisans Tezi olarak kabul edilmiştir.

Danışman : Prof.Dr. M. Kemal YOĞURTÇUGİL İmza

Jüri Üyesi : Prof.Dr. Nalan CİNEMRE İmza

Jüri Üyesi : Yrd.Doç.Dr. Sezai MAKAS İmza

VERİ MADENCİLİĞİ

Elif Özge Özdamar
Mimar Sinan Üniversitesi, Fen Bilimleri Enstitüsü, Uygulamalı İstatistik Anabilim Dalı

ÖZET

Bu çalışmada amaç, son yıllarda geniş kullanım alanı olan veri madenciliğinde kullanılan teknikleri tanıtmak ve uygulamalarını yapmaktır.

Bu amaç doğrultusunda çalışmanın ikinci bölümünde veri madenciliğinde sınıflandırmaya, üçüncü bölümünde veri madenciliğinde kullanılan kümeleme tekniklerine, dördüncü bölümünde yapay sinir ağlarına, beşinci bölümünde genetik algoritmalara ve altıncı bölümünde görsel veri madenciliğine değinilmiştir.

Yedinci bölümde ise IBM – İntelligent Miner programı kullanılarak satışlara yönelik veri madenciliği analizleri yapılmıştır.

Anahtar kelimeler: Veri madenciliği, yapay sinir ağları, kümeleme, görsel veri madenciliği, genetik algoritmalar

Danışman: Prof.Dr. M. Kemal YOĞURTÇUGİL, Mimar Sinan Üniversitesi, İstatistik Bölümü, Uygulamalı İstatistik Anabilim Dalı

ABSTRACT

The main purpose of this study is, to introduce and to apply the data mining techniques which has wide usage at recent years.

In the light of this purpose, in the second part of this study classifiying in data mining, clustering in data mining in the third part, artificial neural networks in the fourth part, genetic algorithms in the fifth part and visiual data mining in the sixth part has been examined.

And lastly in the seventh part, with the use of IBM- Intelligent Miner, data mining analysis has been applied to sales.

Keywords: Data mining, artificial neural networks, clustering methods, visual data mining, genetic algorithms

Advisor: Prof.Dr. M. Kemal YOĞURTÇUGİL, Mimar Sinan University, Department of Statistics, Application of Statistic Section

TEŞEKKÜR

Üzerimde emeği ve bilgisi olan tüm hocalarıma, Sn. Bölüm Başkanınız Prof. Dr. M. Kemal Yoğurtçugil ve düzeltmelerde yardımcı olan Sn. Prof. Dr. Nalan Cinemre'ye, hayatı yaşanır kılan Eylem Can ve Elif Can' a, ve onsuz bu çalışmayı bitiremeyeceğime inandığım dost Nuray Baş'a...

Tüm kalbimle...

İÇİNDEKİLER

ŞEKİL DİZİNİ

TABLO DİZİNİ

GİRİŞ
VERİ MADENCİLİĞİ

Yapılan bir tahmine göre, birkaç yıl içerisinde, dünya üzerindeki tüm veritabanlarında tutulan veriler, insanlık tarihi boyunca üretilen verilerden çok daha fazla olacaktır. Fiziksel olarak çok yüksek kapasiteli veri depolama elemanlarının geliştirilmesi, bu sorunun giderilmesinde çok büyük bir etken oynamayacaktır.

Özellikle internet üzerinden yapılan her türlü işlemlerde, iletişim, bankacılık ve sigortacılık sektörlerinde, günlük yapılan alışverişlerin tutulduğu market zincirlerinde bir gün içerisinde 'sayılamayacak' kadar veri depolanmaktadır. Veri kirliliği denilen bu durum, araştırmacıları çok büyük veritabanları üzerinde analiz teknikleri geliştirmeye yönlendirmiştir.

90'lı yılların başında ilk kez telaffuz edilmeye başlanan veri madenciliği, büyük veritabanları üzerinde, verilerin depolanmasıyla başlayan ve analiz sonrası sunumuyla sonlanan bir süreç olarak tanımlanmaktadır. Bu sürecin yaklaşık %80'i veri depolama, düzenleme, normalize etme ve değişken seçimi gibi önişlemlerle geçmektedir. Geri kalan zaman ise analize ve sonuçların sunumuna ayrılan zamandır. Bu sürecin analiz kısmı veri madenciliği, tüm süreç ise veritabanlarında bilgi keşfi (knowledge discovery in databases) olarak adlandırılmaktadır.

Veritabanlarının büyümesiyle, sorgu ve OLAP (Online Analytical Process) gibi veritabanları üzerinde uygulanan teknikler yetersiz kalmaya başlamıştır. Bu yüzden sık sık hatta eşanlı olarak güncellenen veritabanlarındaki veriler, belirli periyotlarla temizlenip düzenlenerek veriambarları denilen, veritabanları gibi sıksık güncellenmeyen yüksek kapasiteli bilgisayarlar üzerinde taşınırlar.

Veri ambarları üzerinde istatistik, makine öğrenimi ve bilgiayar bilimlerince geliştirilen analizler uygulanır.

Veri madenciliği istatistiğin tersine, tümdengelim yaklaşımını kullanır. Verilerden farkedilemeyen ve öngörülemeyen bilgi çıkartmaya çalışır. Bu yüzden hipotezler üzerinden modeller kurmak yerine, örüntü (pattern) denilen kesin olmayan fakat varlığı önsel olarak tahmin edilen yapıları araştırır.

Veri madenciliği uygulamaları;

Pazarlama alanında;
- Müşterilerin satın alma örüntülerinin belirlenmesi,
- Müşterilerin demografik özellikleri arasındaki bağıntıların bulunması,
- Posta kampanyalarında cevap verme oranın arttırılması,
- Mevcut müşterilerin elde tutulması, yeni müşterilerin kazanılması,
- Pazar sepeti analizi (Market Basket Analysis),
- Müşteri ilişkileri yönetimi (Customer Relationship Management),
- Müşteri değerlendirme (Customer Value Analyses),
- Satış tahmini (Sales Forecasting).

Bankacılıkta;
- Farklı finansal göstergeler arasında gizli kalmış korelasyonların bulunması,
- Kredi kartı dolandırıcılıklarının tespiti,
- Kredi kartı harcamalarına göre müşteri gruplarının belirlenmesi,
- Kredi taleplerinin değerlendirilmesi.

Sigortacılıkta;
- Yeni poliçe talep edecek müşterilerin tahmin edilmesi,
- Sigorta dolandırıcılıklarının tespiti,
- Riskli müşteri örüntülerinin belirlenmesinde kullanılır.

Bu çalışmada ikinci bölümde sınıflandırma teknikleri, üçüncü bölümde kümeleme teknikleri, dördüncü bölümde yapay sinir ağları, beşinci bölümde genetik algoritmalar, altıncı bölümde görsel veri madenciliği anlatılmıştır. Yedinci bölümde bir şirketin satış departmanı veritabanı üzerinde IBM Intelligent Miner programı kullanılarak kümeleme, yapay sinir ağlarıyla kümeleme, karar ağacı ve bu programın görsel teknikler açısından zayıf olması nedeniyle sadece Treemap uygulaması yapılmıştır.

İKİNCİ BÖLÜM
VERİ MADENCİLİĞİNDE KULLANILAN SINIFLANDIRMA TEKNİKLERİ

Veri madenciliğinde kullanılan sınıflandırma teknikleri ağırlıklı olarak makine öğrenimi disiplinine ait tekniklerdir. Özellikle veri madenciliğinin bir uygulama dalı olan sepet analizinde, sınıflandırma kurallarının farklı bir türü olan birliktelik kuralı kullanılmaktadır.

Sınıflandırma teknikleri veri madenciliğinde doğruluk, hız, anlaşılabilirlik ve öğrenme zamanına bağlı olarak değelendirilirler.

2.1. Sınıflandırma Teknikleri

İstatistik, makine öğrenimi ve yapay sinir ağlarının kullandıkları sınıflandırma teknikleri farklıdır. Veri madenciliğinde, makine öğrenimi tekniklerinden karar ağaçları ve bir çeşit yapay sinir ağı olan SOM sınıflandırma problemlerinde sıkça kullanılmaktadır. Bu disiplinlerin kullandıkları sınıflandırma teknikleri genel olarak açıklandıktan sonra karar ağaçları üzerinde detaylı bir şekilde durulacaktır.

2.1.1. İstatistikte kullanılan sınıflandırma teknikleri

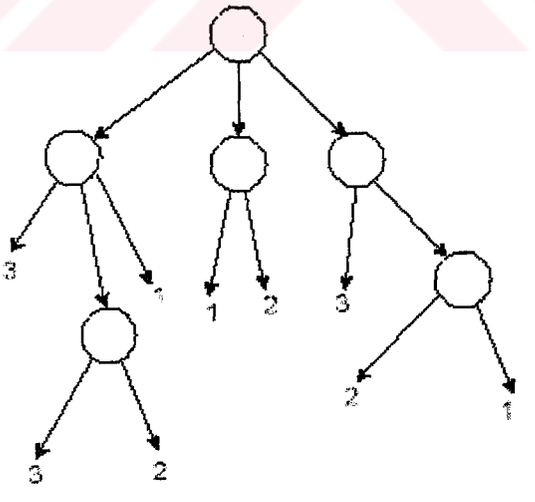
İstatistikte kullanılan sınıflandırma teknikleri klasik ve modern teknikler olmak üzere iki ana başlıkta incelenirler. Klasik teknikler; Fisher (1936) tarafından ortaya atılan ve lineer diskriminant kuralına dayanan tekniklerdir. Bu kural iki farklı yolla; en küçük kareler ya da maksimum olabilirlik kullanarak sınıfları, aralarındaki uzaklık maksimum olacak şekilde ayırır. Bu kurala dayanılarak daha sonra quadratik ve lojistik diskriminant fonksiyonları geliştirilmiştir. Klasik sınıflandırma tekniklerinden bir diğeri ise Bayes kuralıdır.

İstatistikte sınıflandırma amaçlı kullanılan modern tekniklerin en eskisi Fix ve Hodges tarafından geliştirilen ve parametrik olmayan bir teknik olan yoğunluk tahminidir. Diğer modern teknikler; k-en yakın komşu, Projection Pursuit Sınıflandırma, Naive Bayes ve rastlantısal ağlardır. Ayrıca ACE (The Alternatig Conditional Expectation) ve MARS (Multivariate Adaptive Regression Spline) algoritmaları, istatistikçiler tarafından geliştirilmiş sınıflandırma algoritmalarıdır.

2.1.2. Makine öğreniminde kullanılan sınıflandırma teknikleri

Makine öğreniminde kullanılan sınıflandırma teknikleri, sınıflandırma kuralları ve karar ağaçları olmak üzere iki ana başlık altında toplanırlar.

Bir karar ağacı, kök düğümden (root node) başlanarak ağacın dallanmasıyla bir değişkenin test edildiği karar / test düğümlerine ya da ağacın bitiş düğümü olan ve bir sınıfı belli eden yaprak düğümüne ulaşılmasıyla elde edilir. Şekil 2.1.'de bir karar ağacı şeması verilmiştir.



Şekil 2.1.
Karar ağacı şeması

Karar ağaçları aşağıda belirtilen 3 özellikleri nedeniyle veri madenciliğinde tercih edilmektedirler;

1. Gözlemlenmemiş örnekleri genelleştirebilmeleri,
2. Etkili olmaları,
3. Kolay algılanabilir sonuç vermeleri.

Bir karar ağacının boyutu, bir düğümde kaç değişkeni test ettiğine bağlıdır. Birden fazla değişkeni test eden karar ağaçları çok boyutlu karar ağaçları olarak adlandırılırlar ve karar ağacına sınır getirdiklerinden açıklanması zor hatta imkansız olan sınıfları elimine ederler.

Karar ağaçları sayısal verilerin yanısıra sembolik verileri de sınıflandırırlar. Bu özellikleri veri madenciliğinde sıkça tercih edilmelerine neden olur.

Veri setinde kayıp verilerin olması, karar ağaçlarını kurarken problem yaratır. Kayıp verilerin yarattığı problemleri gidermek için bazı teknikler geliştirilmiştir. Kayıp verisi olan bir örneği gözardı etmek, yerine en benzeyen örneği kullanmak, kayıp verilerin alabileceği tüm olası değerleri, olasılıkları oranında birleştirmek bu tekniklere örnek olarak sayılabilir. Karar düğümlerinde birden çok test olması durumunda, kayıp verileri gözardı etmek, anlamlı olan örneklerin sayısını azaltır.

Başka bir teknik ise, örnek ortalamasını kullanarak kayıp veriyi tahmin etmektir. Her düğümde sınıflandırılmış olan veriler ortalamaları 0, standart sapmaları 1 olacak şekilde normalize edilererek kayıp veriler için örnek ortalaması olan sıfır değeri alınır. Bu tekniğin çok boyutlu bir karar ağacında kullanılması durumunda, testlerin beraber kullanımı bir lineer kombinasyon örneği olduğundan, sıfırın etkisinden dolayı bazı değişkenler sınıflandırmaya katılmayabilir.

Veri setinin çok büyük olması durumunda, karar ağaçları çok fazla dallanırlar. Bunu önlemek için karar ağaçları budanır. İki çeşit budama tekniği vardır. İlk teknikte, ağaçtan edinilen bilginin olmaması ya da yetersiz olması durumunda ağacın genişlemesi durdurulur. Bir karar ağacından edinilen bilgi Quinlan tarafından ortaya atılan entropi formülüyle hesaplanır. p_i; t düğümündeki sınıfın gerçekleşme olasılığı olmak üzere; t düğümünün entropisi, p_i düğümdeki sınıfın gerçekleşme olasılığı olmak üzere;

$$\text{entropi}(t) = \sum_{i=1}^{c} -p_i \log_2 p_i \qquad (2.1)$$

şeklinde hesaplanır. n,düğümdeki örnek sayısı olmak üzere bir dalın entropisi ise aşağıdaki gibi formüllenir:

$$\text{entropi}(S) = \sum_{k=1}^{m} \frac{n_k}{n} \sum_{l=1}^{c} -\frac{n_{kl}}{n_k} \log_2 \frac{n_{kl}}{n_k} \qquad (2.2)$$

İkinci tip budama tekniği; bir düğümdeki örnek sayısının tüm veri setine oranlanmasıyla elde edilen örnek oranı, tanımlanan bir eşik değerine (yüzde olarak verilir) ulaşınca ağacın büyümesi durdurulur.

Veri madenciliğinde en çok kullanılan karar ağaçları; C4.5, NewID, AC^2, CART, Cal5, Bayes Tree, CN2 ve Itrule'dur.

2.1.3. Sınıflandırma yapan yapay sinir ağları

Sınıflandırma yapan yapay sinir ağlarına dördüncü bölümde değilecektir. Çok sık kullanılmasalar da Ramnets, DIPOL92 ağları sınıflandırma yapan ağlardır.

ÜÇÜNCÜ BÖLÜM
VERİ MADENCİLİĞİNDE KULLANILAN KÜMELEME TEKNİKLERİ

3.1. Öğrenim Çeşitleri

Farklı disiplinlerce geliştirilmiş birçok kümeleme algoritması vardır. Fakat veri madenciliğinde sıkça başvurulan kümeleme algoritmaları, ağırlıklı olarak makine öğrenimi disiplini tarafından geliştirilmiş olanlardır.

Makine öğrenimi, kullandığı algoritmaları öğrenim yöntemlerine göre sınıflandırır. Kümeleme bir çeşit öğrenim yöntemidir. Makine öğreniminde 3 farklı öğrenim yöntemi vardır:

1. Denetimli öğrenim (Supervised learning): Denetimli öğrenimde amaç; girdi ve çıktı değerlerinin arasındaki ilişkiyi öğrenmektir. Böylelikle yeni bir girdi değeri için bir çıktı değeri tahminlenebilir. Sınıflandırma algoritmaları, özellikle regresyonun dahil olduğu geleneksel istatistik teknikleri, yapay sinir ağları ve destekli vektör makineleri denetimli öğrenime örnektir.

2. Destekli öğrenim (Reinforcement learning): Destekli öğrenim, hayvanların öğrenme şeklini modelleyen öğrenim yöntemidir. Öğrenici eylemde bulunur, eylemi nedeniyle çevresindeki değişimlerden sadece geribesleme (feedback) ile bilgi alır. Dinamik kaynak paylaşımı (dynamic resource allocation), oyun oynama (game playing) ve geçici uyuşmazlık öğrenimi (temporal difference learning) destekli öğrenime örnektir.

3) Denetimsiz öğrenim (Unsupervised learning): Denetimsiz öğrenimde amaç, girdi değerlerine (verilere) en uygun gösterim şeklini bulmaktır. Birçok farklı yaklaşım vardır. Bilgi maksimizasyonu, minimum çapraz entropi, minimum yenidenyapılandırma hatası gibi. Veri sıkıştırma, dağılım tahmini, veri kaynağının ayırımı, veri görselleştirme gibi denetimsiz öğrenimin çeşitli

uygulamaları, denetimli öğrenim için önişlemler (preprocess) olarak kullanılabilirler. Kümeleme, en temel denetimsiz öğrenim yöntemidir. (Grapel, 1998)

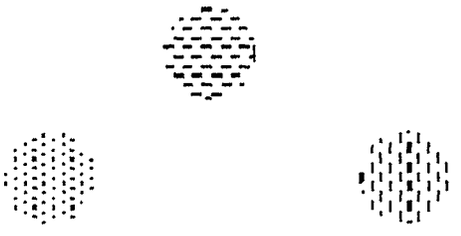
Kümeleme algoritmalarının veri madenciliğinde kullanılabilmeleri için; ölçülebilirlilik, çok yönlülük, farklı yapıdaki kümeleri bulabilme, gereken girdi parametrelerinin minimum olması, kayıp verilerden çok etkilenmeme, verilerin sıralarının değişmesi durumuna duyarlı olmama ve çok boyutlu verilerde rahatça çalışabilme gibi özelliklere sahip olması gerekir. (Zaiane, Foss, Lee and Wang, 2002)

3.2. Küme ve Kümeleme Tanımları

Nesnelerin kendilerini ya da diğer nesnelerle olan ilişkilerini tarif eden bilgileri kullanarak nesneleri gruplara ayırma işlemine kümeleme denir. Kümelemede amaç; grup içindeki nesneleri, diğer gruplardaki nesnelerden olabildiğince ayrı/bağımsız, kendi aralarında ise birbirine benzer/bağımlı olacak şekilde oluşturmaktır. (Zaiane, 2000)

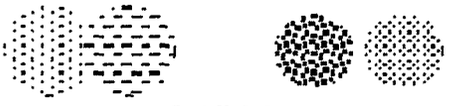
Farklı kümeleme algoritmaları, farklı küme tanımlarına dayanır. 5 temel küme tanımı vardır:

1. İyi ayrılmış küme tanımı: Bir küme içindeki herhangi bir nokta, o küme içindeki diğer tüm noktalara, küme dışındaki herhangi bir noktadan daha yakındır (ya da benzerdir).



Şekil 3.1.
İki boyutlu 'iyi ayrılmış' kümeler
(Zaiane, 2000)

2. Merkeze dayalı küme tanımı: Bir küme içindeki her nokta, o kümenin merkezine, diğer kümelerin merkezlerine olduklarından daha yakındır (ya da benzerdir). Bir kümenin merkezi, centroid ya da medoid gibi o kümeyi temsil eden bir noktadır.

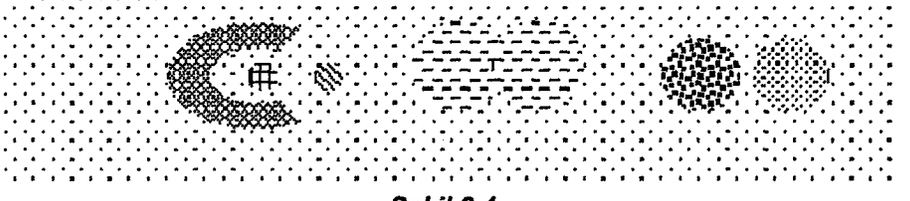


Şekil 3.2.
İki boyutlu 'mekeze dayalı' kümeler
(Zaiane, 2000)

3. En yakın komşu küme tanımı: Bir küme içindeki herhangi bir nokta, küme içindeki diğer bir noktaya ya da noktalara başka bir kümedeki noktalardan daha yakındır (ya da benzerdir).

Şekil 3.3.
İki boyutlu 'en yakın komşu' kümeleri
(Zaiane, 2000)

4. Yoğunluğa dayalı küme tanımı: Bir küme, noktaların yoğunluklarına göre diğerlerinden ayrılır. Yüksek yoğunluklu alanlar, düşük yoğunluklu alanlar tarafından ayrılır. Veri setinde gürültü ve aykırı gözlem olması durumunda bu tanım önem kazanır.



Şekil 3.4.
İki boyutlu 'yoğunluğa dayalı' kümeler
(Zaiane, 2000)

5. Benzerliğe dayalı küme tanımı: Bir küme, birbirine benzer nesnelerden oluşur. Kendi dışındaki kümelerin nesneleri, kendi nesnelerine benzemez. Bu tanım özellikle noktaların oluşturdukları yoğunluk ya da şekle göre kümeleri farklı olarak tanımlar.

3.3. Kümeleme Teknikleri

Veri madenciliğinde kullanılan kümeleme teknikleri;

- Merkeze dayalı ayırıcı kümeleme teknikleri
- Hiyerarşik kümeleme teknikleri
- Yoğunluğa dayalı kümeleme teknikleri
- Grafik esaslı kümeleme teknikleri

olmak üzere 4 ana başlıkta toplanır.

3.3.1. Merkeze dayalı ayırıcı kümeleme teknikleri

Ayırıcı kümeleme teknikleri, sezgisel teknikler olarak da bilinirler. Optimal bir kritere göre iteratif olarak veri setini bölümlere ayıran tekniklerdir. (Zaiane, Foss, Lee and Wang, 2002)

Birçok ayırıcı kümeleme tekniği olmasına karşın bu tür algoritmalar birbirine oldukça yakın sonuç verirler. En sık kullanılan ayırıcı kümeleme teknikleri K-means ve K-medoid yaklaşımlarıdır.

Her iki yaklaşım da bir kümenin ağırlık merkezinin (gravity center) o kümeyi temsil ettiği iddiası üzerine kurulmuştur. K-means; kümenin ağırlık merkezi olarak o kümenin ortalamasını ya da medyanını alır ve merkezi nokta (centroid) olarak kabul eder. Anlaşılacağı gibi centroid gerçek bir nokta değildir. K-medoid ise ağırlık merkezi olarak bir kümeyi en iyi şekilde temsil eden nokta olan medoid kavramına dayanır. Medoid, kümenin orta noktasıdır ve gerçek bir noktadır. Centroid ve medoid aynı zamanda temsil noktaları (representitive points) olarak da adlandırılırlar.

Ayırıcı kümeleme yapan algoritmalar, ağırlık merkezi olarak tek bir noktayı ele aldıklarından ve kümeleri, noktaların ağırlık merkezine olan bağıl uzaklıklarına göre oluşturduklarından küre şeklindeki kümeleri bulabilirler. Ayırıcı algoritmaların diğer dezavantajları aşağıdaki gibi özetlenebilir.

- Bulunacak (k) küme sayısının önceden bilinmesi zorunluluğu,
- Büyük boyutlu kümelerin, algoritma gereği bölünmesi yüzünden tanımlanmasının zor olması ve
- Sadece konkav küre kümeler için uygun olmalarıdır.

3.3.1.1. K-means kümeleme algoritması

K-means kümeleme algoritması oldukça basit bir algoritmadır. Bu algoritmayı oluşturan adımlar;

1. K tane başlangıç merkezi noktası (centroid) seçilmesi,
2. Tüm noktaların en yakındaki merkezi noktaya atanması,
3. Her küme için merkezi noktaların yeniden hesaplanması,
4. 2. ve 3. adımları merkezi noktalar değişmeyinceye kadar tekrarlanması

şeklinde açıklanabilir.

K-means algoritmasına örnek olarak, kategorik verilerin kümelenmesinde kullanılan, her kümenin ortalaması yerine modunu ağırlık merkezi olarak ele alan k-modes metodu verilebilir.

K-means algoritmasına başka bir örnek de Bisecting K-means algoritmasıdır. Bu algoritma; veri setindeki en büyük kümeyi bularak, bunu genel K-means algoritmasıyla ikiye böler. İkiye bölme işlemi, istenilen küme sayısına erişilinceye kadar devam eder. Bu algoritmanın genel K-means algoritmasından daha farklı boyutta kümeler oluşturması, düşük entropiye yol açar (Zaiane, Foss, Lee and Wang, 2002).

K-means üzerinde yapılan son çalışmalar, kümelerin bölümlendirilmesinde sayısal değerler için sınıfiçi süredurum (intraclass inertia), kategorik değerler için de yeni condorcet ölçütlerinin ve farklı bir uzaklık kavramının ortaya atılması üzerine yoğunlaşmıştır. (Gutta, Sabbasiva Raok and Bhatnagar, 1999)

K-means algoritmasıyla veri setinin kümelere ayrılmasında, bilgisayar tarafından kullanılan alan, m nokta sayısı, n de değişken sayısı olmak üzere O(mn) olarak gösterilir. Sonuca ulaşmak için gereken zaman ise O(I*K*m*n) dir. Burada I; tüm kümeler bulununcaya kadar yapılan iterasyon sayısıdır ve genellikle 5 ile 10 arasındır. (Zaiane, 2000)

Başlangıç merkezi noktalarının seçilmesi

Başlangıç merkezi noktalarının doğru olarak seçilmesi, K-means algoritmasının en temel aşamasıdır. Başlangıç merkezi noktalarını rassal olarak seçmek kolay ve etkili olabilir, fakat sonuçları her zaman tatmin edici değildir. Bunun yerine rassal olarak farklı başlangıç merkezi noktalarından oluşan kümeler ele alınarak K-means algoritması tekrar tekrar çalıştırılabilir. Bazı çalışmalar 30 ayrı nokta kümesi önerir, fakat bu yöntem de veri setine göre ve araştırılan küme sayısına göre tatmin edici sonuçlar vermeyebilir. Şekil 3.5.'de 10 küme için seçilen başlangıç merkezi noktaları gösterilmektedir. Başlangıç noktaları hatalı seçildiğinden, K-means algoritması sonucu optimal değildir (Şekil 3.6.).

şekil 3.5.
(Zaiane, 2000)

şekil 3.6.
(Zaiane, 2000)

Rasgele örnekleme tüm kümeleri kapsamayabileceğinden, başlangıç merkezi noktaları seçiminde başka yöntemlere başvurulabilir. Başlangıç merkezi noktaları veri setinin özellikle sıkışık alanlarından seçilir ve bu yolla noktaların iyi ayrılmaları sağlanmış olur. Böylelikle tek bir kümeden iki tane ayrı merkezi nokta seçilmesinin de önüne geçilmiş olunur.

Merkezi noktaların eklemeli olarak güncellenmesi

Merkezi noktaların, tüm noktaların kümelere atanma işlemi bittikten sonra bir kerede güncellenmeleri yerine, her bir noktanın bir kümeye atanmasından sonra her defasında güncellenmeleri daha kesin ve hızlı bir sonuç verir. Merkezi noktaların eklemeli olarak güncellenmesi denilen bu işlem, bütün noktaların kümelere atanmasına kadar devam eder ve yapay sinir ağlarında ağırlıkların güncellenmesine benzer.

Başlangıç merkezi noktalarının iyi seçilmemesi durumunda ve tüm noktaların kümelere atanmasından sonra boş kümelerin oluşması sıkça rastlanan bir durumdur. Eklemeli güncelleme; kümeleme işlemini hızlandırmak ve daha iyi sonuçlar vermesini sağlamanın yanısıra boş (varolmayan) kümelerin oluşmamasını sağlar. Bu teknik yanlışlıkla boş bir küme için başlangıç merkezi noktası seçilmesi durumunda, her bir nokta için merkezi noktalar güncellendiğinden, boş kümeye yeni bir merkezi nokta belirleyecektir. Aksi durumda hata kareleri olması gerekenden çok daha büyük çıkacaktır.

Eklemeli güncellemede en çok kullanılan yaklaşım, yeni merkezi nokta olarak varolan merkezi noktaya en uzaktaki noktanın alınmasıdır.

Merkezi noktaların tanımları

K-means algoritması veri setini; noktaların, içinde bulundukları kümenin orta noktasına olan uzaklıklarının karesi minimum olacak şekilde k tane kümeye ayırma problemidir. Bu da hatanın minimize edilmesi demektir.

$\vec{X}$ nokta gösteren vektör, $\vec{C}_i$ küme merkezi noktası, d; $\vec{X}$ ve $\vec{C}_i$' nin boyutu, x_j ve c_{ij} de bileşenleri olmak üzere hata;

$$\text{Hata} = \sum_{i=1}^{K} \sum_{\vec{x} \in C_i} \left\| \vec{x} - \vec{c}_i \right\|^2 = \sum_{i=1}^{K} \sum_{\vec{x} \in C_i} \sum_{j=1}^{d} \left(x_j - c_{ij} \right)^2 \tag{3.1}$$

şeklinde gösterilir.

p. kümenin ($\vec{C}_p$), her bileşeni (c_{pk}) için ($1 \le k \le d$) hatanın diferansiyeli alınıp 0' a eşitlenmesi durumunda;

$$\frac{\partial}{\partial c_{pk}} \text{Hata} = \frac{\partial}{\partial c_{pk}} \sum_{i=1}^{K} \sum_{\vec{x} \in C_i} \sum_{j=1}^{d} \left(x_j - c_{ij} \right)^2 \tag{3.2}$$

$$= \sum_{i=1}^{K} \sum_{\vec{x} \in C_i} \sum_{j=1}^{d} \frac{\partial}{\partial c_{pk}} \left(x_j - c_{ij} \right)^2 \tag{3.3}$$

$$= \sum_{\vec{x} \in Cp} 2\left(x_k - c_{pk} \right) = 0 \tag{3.4}$$

elde edilir. Bu denklemden c_{pk} çekildiğinde

$$\sum_{\vec{x} \in C_p} 2\left(x_k - c_{pk} \right) = 0 \Rightarrow n_p c_{pk} = \sum_{\vec{x} \in C_p} x_k \Rightarrow c_{pk} = \frac{1}{n_p} \sum_{\vec{x} \in C_p} x_k \tag{3.5}$$

kümenin içindeki noktaların ortalaması bulunur;

$$\vec{C}_p = \frac{1}{n_p} \sum_{\vec{x} \in C_p} \vec{X} \tag{3.6}$$

Bu formül aynı zamanda kümenin orta noktasının nerede olması gerektiğini gösterir.

Her noktanın küme merkezine olan toplam uzaklığının karesi yerine toplam uzaklığı ele alınırsa;

$$\text{Hata} = \sum_{i=1}^{K} \sum_{\vec{x} \in C_i} \left\| \vec{X} - \vec{C}_i \right\| \tag{3.7}$$

olur. Yöneylem araştırmasında çok-kaynak Weber problemi olarak geçen bu yaklaşım, genellikle veri ambarlarının bir orta noktaya yerleştirilmesinde kullanılır. Minimize edilecek hata;

$$\text{Hata} = \sum_{i=1}^{K} \sum_{\vec{x} \in C_i} \sum_{j=1}^{d} \left| x_j - c_{ij} \right| \tag{3.8}$$

şeklinde yazılabilir. Hatayı minimize etmek için p. kümenin ($\vec{C}_p$), her bileşeni (c_{pk}) için ($1 \le k \le d$) hatanın diferansiyeli alınıp sıfıra eşitlenir.

$$\frac{\partial}{\partial c_{pk}} \text{Hata} = \frac{\partial}{\partial c_{pk}} \sum_{i=1}^{K} \sum_{\vec{x} \in C_i} \sum_{j=1}^{d} \left| x_j - c_{ij} \right| \tag{3.9}$$

$$= \sum_{i=1}^{K} \sum_{\vec{x} \in C_p} \sum_{j=1}^{d} \frac{\partial}{\partial c_{pk}} \left| x_i - c_{ij} \right| \tag{3.10}$$

$$= \sum_{\vec{x} \in Cp} \sum_{j=1}^{d} \frac{\partial}{\partial c_{pk}} \left| x_k - c_{pk} \right| = 0 \tag{3.11}$$

$$= \sum_{\vec{x} \in Cp} \sum_{j=1}^{d} \frac{\partial}{\partial c_{pk}} \left| x_k - c_{pk} \right| = 0 \Rightarrow \sum_{\vec{x} \in Cp} (\pm)(x_k - c_{pk}) = 0 \tag{3.12}$$

Son eşitlik $\vec{C}_p$ için çözüldüğünde; $\vec{C}_p = \text{medyan}(\vec{X} \in C_p)$ elde edilir. $\vec{C}_p$ noktasının koordinatları; kümedeki noktaların koordinat değerlerinin medyanıdır.

İyileştirme işlemleri

Kümelemenin en iyi şekilde gerçekleşmesi için, kümelerin hesaplanmasından önce (pre-processing) ve sonra (post-processing) olmak üzere bazı iyileştirme işlemleri yapılır.

Hata kareleri kriteri kullanıldığında aykırı gözlemlerin kümeleri olumsuz yönde etkilemesi sıkça karşılaşılan bir durumdur. Aykırı gözlemlerin veri setinden çıkartılarak kümelemeye başlanması, kümelerin bulunmasından önce yapılabilecek işlemlere örnektir.

Kümeler bulunduktan sonra yapılan işlemler ise, genellikle aykırı gözlemleri gösterdiklerinden küçük kümelerin gözardı edilmeleri ya da çok büyük kümelerin daha küçük kümelere ayrılmalarıdır.

ISODATA, 80'li yılların ortalarında bulunan, hala kullanılan, özellikle görsel veriler üzerinde yapılan işlemlerde sıkça kullanılan bir iyileştirme işlemidir. Bu yöntem başlangıç centroid'lerinin kümeleri en iyi biçimde ayıracak şekilde seçilmesine garanti verir.

ISODATA nın algoritması

1. Noktalar en yakın centroide atanır ve küme centroidleri yeniden hesaplanır Bu işlem noktalar kümeleri değiştirmeyinceye kadar devam eder.
2. "Çok az " nokta içeren kümeler gözardı edilir.
3. Kümeler birleştirilir ya da ayrılır;

a. Çok fazla küme varsa, kümeler ayrılır.

b. Tahmin edilenden çok daha az küme varsa, kümeler birleştirilir.

c. Kümeleri birleştirmeye ya da ayırmaya alternatif olarak: orta noktaları birbirine çok yakınsa kümeler birleştirilirler, çok ayrıksa ayrılırlar.
4. 1, 2 ve 3. adımlara, sonuçlar kabul edilebilir olana kadar ya da önceden belirlenmiş iterasyon sayısına ulaşılıncaya kadar devam edilir.

ISODATA' nın kümeleri ayırması, kümelerin sayısını arttırarak toplam hatayı düşürmeye yöneliktir. ISODATA kümeleri ayırmada 2 yöntem kullanır:

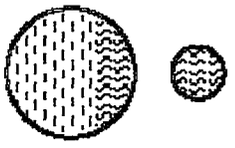
1. Kümeyi bölmek: Genel eğilim, en büyük hataya sahip olan kümeleri bölme yönünde olmakla birlikte ISODATA' da belli bir değişkene ait en büyük standart sapmalı küme bölünür.
2. Kümeye yeni bir orta noktası eklemek: Bölünecek kümenin orta noktasına en uzak olan nokta yeni kümenin orta noktası olarak seçilir.

ISODATA' nın kümeleri birleştirmesi ise kümelerin sayısını azaltmaya yöneliktir, fakat bu işlem hatayı arttırabilir. Optimal bir şekilde kümeleri birleştirmede ISODATA iki yöntem kullanır:

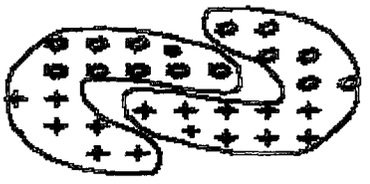
1. Kümeleri yaymak: Hatayı en az arttıran kümedeki noktalar en yakın kümeye atanır.
2. Kümeleri birleştirmek: Merkezi noktaları birbirine en yakın olan kümeler ya da hatayı en az arttıracak olan kümeler birleştirilir. Bu iki yaklaşım da hiyerarşik kümeleme yöntemlerinde kümeleri birleştirmede kullanılan Ward ve centroid yaklaşımlarıyla aynıdır.

Limit ve problemler

K-means, mutlak hatayı ya da hata karelerini, küme merkezi noktalarına göre minimum kılınmasını kriter alır. Bu kritere göre kümeleme, özellikle kümelerin farklı boyutta (Şekil 3.7.) ya da konveks (Şekil 3.8.) olmaları durumunda iyi sonuç vermeyebilir, K-means birbirinden oldukça ayrık kümelerin ya da eşit sayılabilecek boyutlara sahip kümelerin bulunduğu veri setlerinde kullanışlıdır.

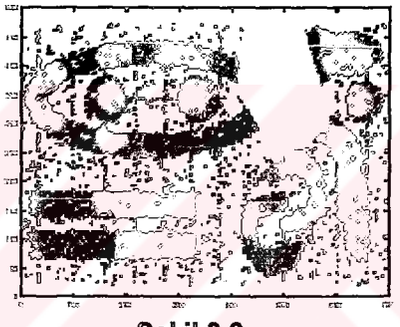


Şekil 3.7.
Farklı boyutta kümeler
(Zaiane, 2000)



Şekil 3.8.
Konveks şekilli kümeler
(Zaiane, 2000)

Şekil 3.9.'da K-means algoritmasıyla kümelere ayrılmış bir veri seti gösterilmiştir. Algoritma veri setini 9 kümeye ayırmıştır. (k=9)



Şekil 3.9.
K-means
(Zaiane, Foss, Lee and Wang, 2002)

3.3.1.2. K-medoid kümeleme algoritması

K-medoid algoritması, her bir kümeyi en iyi şekilde temsil edecek noktaları kullanarak kümeleri üstüste gelmeyecek şekilde oluşturur. Bu noktalar uzaklık gibi bir ölçü değerine bağlı olarak kümelerin ortasında bulunan noktalardır. K-medoid algoritmasının adımları aşağıda açıklanmıştır.

K-medoid algoritması:

1. K tane başlangıç noktası seçilir. Kendi kümelerinin en orta noktaları oldukları varsayılan bu noktalar aday medoidlerdir.
2. Seçilen bu noktaların ayrı ayrı seçilmemiş bir noktayla yer değiştirmesi durumu incelenir. Bu şu şekilde olur: Seçilmeyen her noktanın en yakın aday medoide olan uzaklığı hesaplanır. Toplam uzaklık maliyet olarak adlandırılır. Daha sonra seçilmiş olan nokta başka bir seçilmemiş bir noktayla değiştirilir

ve maliyet hesaplanır. Tüm olası değişimler için maliyetlerin hesaplanmasından sonra 3. adıma geçilir.

3. En düşük maliyetli durum seçilir.

4. En düşük maliyetli durum seçilmemiş bir noktayla sağlanıyorsa, nokta en yakın seçilen nokta olan medoide atanır.

3.3.1.3. CLARANS

CLARANS, veri madenciliği için geliştirilmiş ilk kümeleme algoritmalarından biridir. İstatistikte kullanılan kümeleme algoritmaları PAM (Partitioning Around Medoids) ve CLARA (Clustering LARge Applications) ya dayanır. PAM, 4 aşamalı K-medoid algoritmasını uygular. CLARA ise PAM'ın büyük veri setlerine uyacak şekilde geliştirilerek uyarlanmış halidir. Veri setinden örneklemeyle nokta seçerek bu noktaların medoidlerini hesaplar, maliyeti minimize eden medoid kümesini seçer. Algoritmanın optimizasyonu için, bir iterasyon sonrası elde edilen medoid kümesi, bir sonraki iterasyon için örnek küme oluşturur. CLARA genellikle 40+5K tane medoid seçer ve 5 iterasyon sonucu veri setini kümelere ayırır.

CLARANS' ın algoritması:

1. Rassal olarak K tane aday medoid seçilir.

2. Rassal olarak seçilmiş noktalardan bir tanesinin seçilmemiş olanlardan bir tanesiyle değiştirilme durumu irdelenir.

3. Yeni durum daha iyi (daha düşük maliyetli) ise 2. adım yeni durumdan başlanarak tekrar edilir.

4. Yeni durum öncekinden daha kötü ise 2. adım varolan durumla tekrar edilir. Tekrar etme sayısı maksimum (250, K(m-k)) olacak şekilde bir parametredir.

5. Ulaşılan son durum daha önceki durumla karşılaştırılır.

6. 2. adımda optimal durum kararı alınmamışsa 1. adıma geri dönülür.

3.3.2. Hiyerarşik kümeleme teknikleri

Hiyerarşik kümeleme teknikleri, toplayıcı ve ayırıcı olmak üzere iki şekilde tasarlanır. Hiyerarşik kümeleme teknikleri, yakınlık matrisini (proximity matrice) kullandıklarından harcadıkları zaman ve alan diğer kümeleme tekniklerine göre biraz daha fazladır.

Hiyerarşik kümeleme teknikleri, m veri sayısı olmak üzere m^2 tane yakınlık hesapladıklarından, hesaplanan değerlere yetecek kadar alan ayrılır. İşlemler için gereken süre $O(m^2)$ şeklinde ifade edilir.

3.3.2.1 Ayırıcı (divisive) kümeleme teknikleri

Toplayıcı kümeleme tekniklerinin tam tersi şekilde işlerler. Bu tekniklerde her adımda hangi kümenin ayrılacağına karar verilir. Ayırıcı kümeleme teknikleri pek sık tercih edilmezler.

Ayırıcı kümeleme tekniklerin algoritması

Ayırıcı kümeleme tekniklerinden Minimum Yayılmalı Ağaç (MST; Minimum Spanning Tree)' ın algoritması:

1. Yakınlık grafiği için minimum yayılmalı ağaç hesaplanır.
2. En büyük uzaklığa (en az benzerliğe) göre kümeler arası varolan bağıntılar ayrılarak yeni kümeler oluşturulur.
3. Tüm kümeler singleton (tek noktadan oluşan küme) kümeler oluncaya kadar 2. adım tekrarlanır.

3.3.2.2 Toplayıcı (agglomerative) kümeleme teknikleri

Aşağıdan yukarıya doğru dizayn edilir. Her bir veri, en küçük küme olarak kabul edilir. En altta veri sayısı kadar küme vardır. Yukarıya doğru çıkıldıkça küçük kümeler birleşerek daha büyük kümeler oluştururlar. En üstte ise tüm verileri, dolayısıyla tüm kümeleri kapsayan tek bir küme vardır.

Toplayıcı kümeleme tekniklerinin algoritması

Toplayıcı kümeleme tekniklerinin birbirinden farklılaştığı durum, iki küme arasındaki yakınlığı (proximity) hesaplama biçimlerinden kaynaklanır. MIN, MAX, Group Average gibi toplayıcı kümeleme tekniklerinde kullanılan yakınlık kavramı, Lanc-Williams formülüne dayalı yakınlık kavramıdır.

$$p(R,Q) = \alpha_A p(A,Q) + \alpha_B p(B,Q) + \beta p(A,Q) + \gamma | p(A,Q) - p(B,Q) | \qquad (3.13)$$

Bu formül R ve Q kümeleri arasındaki yakınlığı verir. R kümesi, A ve B kümelerinin birleşmesinden oluşur. Başka bir deyişle; A ve B kümeleri R kümesini oluşturmak için birleştirildiğinde, oluşan yeni kümenin (R) varolan başka bir kümeye (Q) olan uzaklığı, varolan kümenin (Q) orijinal kümeler olan A ve B ye olan uzaklığının doğrusal bir fonksiyonudur. (Zaiane, 2000)

MIN – Single Link

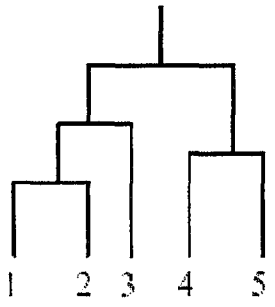
MIN kümeleme tekniği, iki küme arasındaki yakınlığı; bu iki küme içindeki herhangi iki noktanın birbirine uzaklığı minimum olacak şekilde tanımlar. Uzaklığın minimum olması, benzerliğin maksimum olması demektir.

MIN aynı zamanda Sigle Link olarak da geçer. Bunun nedeni, her bir noktayı tek noktadan oluşan küme olarak ele alıp, güçlü linklere öncelik tanıyacak şekilde noktaları kümelere linklerle bağlamasıdır. MIN eliptik olmayan şekildeki kümeleri iyi ayırır fakat gürültü ve aykırı gözlemlere karşı duyarlıdır.

	I1	I2	I3	I4	I5
I1	1.00	0.90	0.10	0.65	0.20
I2	0.90	1.00	0.70	0.60	0.50
I3	0.10	0.70	1.00	0.40	0.30
I4	0.65	0.60	0.40	1.00	0.80
I5	0.20	0.50	0.30	0.80	1.00

şekil 3.10.
Benzerlik matrisi
(Zaiane, 2000)

Şekil 3.10.'da gösterilen benzerlik matrisinin MIN algoritmasıyla kümelere ayrılmasıyla oluşan dendogram Şekil 3.11.'de verilmiştir.

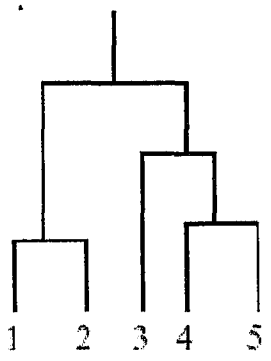


Şekil 3.11.
MIN
(Zaiane, 2000)

MAX – Complete Link – CLIQUE

MAX kümeleme tekniği, iki küme arasındaki yakınlığı; bu iki küme içindeki herhangi iki noktanın birbirine uzaklığı maksimum olacak şekilde tanımlar. Uzaklığın maksimum olması, benzerliğin minimum olması anlamına gelir.

MAX aynı zamanda Complete Link ya da en bilinen adıyla CLIQUE olarak da geçer. Algoritması MIN' e benzer. Tek fark, tüm noktalar bağlanmadan noktaların küme oluşturamamasıdır. Şekil 3.12.'de daha önce kullanılan benzerlik matrisinin MAX algoritması kullanılarak oluşturulan dendogramı verilmiştir.



Şekil 3.12.
MAX
(Zaiane, 2000)

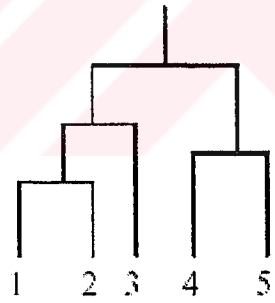
Group Average

Group Average, MIN ve MAX' ın geliştirilmiş şeklidir. Bu algoritma veri setini iki küme arasındaki yakınlık; iki kümedeki bütün noktaların yakınlıklarının ortalaması olacak şekilde kümelere ayırır. Group Average' ın kullandığı yakınlık;

$$\text{yakınlık(küme1,küme2)} = \frac{\sum_{\substack{p_1 \in \text{Küme1} \\ p_2 \in \text{Küme2}}} \text{yakınlık}(p_1, p_2)}{\text{büyüklük(küme1)} * \text{büyüklük(küme2)}} \qquad (3.14)$$

şeklindedir.

Şekil 3.13.'de daha önce kullanılan benzerlik matrisinin Group Average algoritması ile oluşturulan dendogramı verilmiştir. MIN ile aynı sonucu vermiştir.



Şekil 3.13.
Group Average
(Zaiane, 2000)

Ward Yöntemi

Ward yöntemi; iki küme arasındaki yakınlığı K-means algoritmasında kullanılan kritere göre tanımlar. Yakınlık kavramı; iki kümenin birleşmesi sonucunda hata karesindeki artış olarak kabul edilir ve kümeler, hata fonksiyonunun mimimum olmasını sağlayacak şekilde birleştirilir. Bu yaklaşım, karar ağaçlarının oluşturulmasında kullanılan yaklaşıma oldukça benzer. Ward yöntemi, veri setindeki tüm noktaları kullandığı için diğer

toplayıcı kümeleme algoritmalarına göre daha doğru bir hiyerarşik kümeleme gerçekleştirir.

i ; k kümesine ait bir nokta,
x_{ij} ; i noktasının j değişkeni için aldığı değer,
μ_{kj} ; k kümesindeki j değişken değerlerinin ortalaması olmak üzere, E hata fonksiyonu;

$$E = \sum_{k} \sum_{i \in k} \sum_{j} \left(x_{ij} - \mu_{kj} \right)^2 \qquad (3.15)$$

şeklinde formüllenir.

Group Average yönteminde, iki nokta arasındaki yakınlık noktalar aralarındaki uzaklığın karesi olarak alındığında, Ward metoduna çok yakın bir sonuç elde edilir. Ward yönteminin algoritması aşağıda açıklandığı gibidir.

1. p ve q iki nokta olmak üzere, olası tüm çiftler için uzaklık kareleri matrisi oluşturur. Matrisin oluşturulmasında kullanılan ifade şöyledir.

$$d_{pq}^2 = \sum_{j} \left(x_{pj} - x_{qj} \right)^2 \qquad (3.16)$$

2. E hata fonksiyonu minimum olacak şekilde 2 farklı nokta çifti birleştirilir. (d_{pq}^2 değeri minimum olan 2 farklı çift birleştirildiği zaman hata fonksiyonu da minimum olacaktır.)
3. Her d_{pk}^2 için E_{pk} değerleri hesaplanır. E_{pk}, her çiftin (p ve q) k (küme ya da çift) ile birleştiğinde oluşan hata fonksiyonudur.
4. 3. adım her çift ya da küme birleştirilirken E mimimum olacak şekilde tekrar edilir.

5. Tüm çiftler bir kümeye atandığında (alt kümelerin birleşmesinden oluşan ana küme) algoritma durur. (Wishart,1998)·

Hiyerarşik kümeleme tekniklerinin dezavantajları; gürültülü veri setleri, kayıp gözlemli veri setleri ve konveks olmayan kümelerin olduğu veri setleri üzerinde kümeleme yaparken sorun yaşamalarıdır. Ayrıca hiyerarşik kümeleme tekniklerinin büyük kümeleri bölme eğilimleri vardır.

3.3.2.3. CURE (Clustering Using Representatives)

CURE; aykırı gözlemler ve küresel ve/veya düzgün olmayan kümelerin olduğu büyük veri setlerinde kullanan bir hiyerarşik kümeleme algoritmasıdır. CURE, kümenin yerini ve biçimini yakalayabilmek için, o kümeyi temsil edecek birden fazla temsil noktası kullanır.

İlk temsil noktası, kümenin ortasına en uzak olan noktadır. Daha sonraki temsil noktaları ise kendinden bir önceki temsil noktasına en uzakta bulunanlardır. Böylelikle temsil noktaları küme içinde dağılmış olur. Nokta sayısı kullanıcı tarafından seçilir. Yapılan çalışmalar, bir küme için 10 tane temsil noktasının yeterli olduğunu göstermektedir.

Seçilen temsil noktaları α faktörüyle kümenin ortasına doğru 'sıkıştırılır'. Bunun yapılmasının nedeni, aykırı gözlemlerin etkilerini azaltmaktır. Aykırı gözlemler merkezden daha uzakta olduklarından, α faktörüyle sıkıştırıldıklarında, temsil noktalarından daha fazla merkeze doğru hareket edeceklerdir. Örnek olarak, merkezden uzaklığı 10 birim olan bir temsil noktası α = 0.7 ile sıkıştırıldığında 3 birim hareket ederken, 1 birim uzaklıktaki nokta 0.3 birim hareket edecektir.

CURE toplayıcı hiyerarşik şemasını kullanır. İki küme arasındaki uzaklık, herhangi iki temsil noktası arasındaki minimum uzaklıktır.

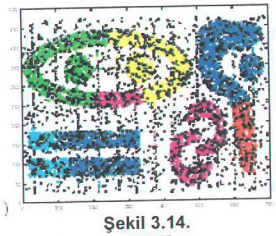
Bir kümenin yavaş yavaş büyüyor olması, onun çoğunlukla aykırı değerler içerdiğine işaret eder. Aykırı değerler merkeze uzakta olduklarından, diğer kümelerle nadiren birleşeceklerdir. CURE' un aykırı değerleri gözardı etmesi için ya küme sayısının nokta sayısının 1/3' ü kadar olması ya da önceden belirlenen k tane kümeden daha fazla küme oluşması yeterli olur. Bu durumda küçük kümeler gözardı edilir.

CURE, karmaşık bir algoritma olduğundan, ($O(m^2 logm)$) çok büyük veri setlerine doğrudan uygulanamaz. Kümeleme hızını arttırmak için CURE, iki yol kullanır. Veri setinden rassal seçimle bir örnek çekerek, hiyerarşik kümelemeyi bu rassal örnek üzerinde uygular. Geri kalan noktaları da oluşan kümelerin en yakın temsil noktasına göre kümelere atar. Veri seti çok büyük olduğunda çekilen örnekler de büyük olacağından, hiyerarşik kümeleme algoritması işlemeyebilir. Bu durumda CURE, ikinci yol olarak veri setini boyutları m/p olan p tane alana ayırır. Her alandaki noktaları m/pq tane küme olana kadar kümelendirir.

CURE algoritması

1. Veri setinden rasgele örnek çekilir.
2. Örneği eşit büyüktükte p parçaya ayırır.
3. Oluşan her parçadaki noktaları, hiyerarşik küme algoritması kullanarak m/pq tane küme olacak şekilde kümelere ayırır. Toplam m/q tane küme oluşur. (Aykırı gözlemlerin bir kısmı bu adımda gözardı edilir.)
4. 3. adımdan artakalan aykırı gözlemler gözardı edilir.
5. Veriler en yakın kümeye atanarak kümeleme tamamlanır.

Şekil 3.14.'de, Şekil 3.9.' da kullanılan veri setinin CURE algoritmasıyla kümelere ayrılması gösterilmiştir. Algoritmaların parametreleri k=0.p ve α=0.3 olarak alınmıştır. Küme başına 10 tane temsil noktası kullanılmıştır.



Şekil 3.14.
CURE
(Zaiane, Foss, Lee and Wang, 2002)

3.3.2.4. Chameleon

Chameleon, iki kümeyi sadece oluşturulacak olan yeni kümenin kendilerine benzediği durumda birleştirir. Chameleon, DBSCAN ve CURE kümeleme algoritmalarına göre uzaysal veriler üzerinde daha iyi sonuç verir.

Chameleon'un ilk adımı yakınlık grafiğini kullanarak k-en yakın komşu grafiğini oluşturmaktır. Yakınlık grafiği, bir nokta ve onun k-en yakın komşusunu, yani o noktaya en yakın ve en çok benzer olan noktaları gösterir. Tüm yakınlık grafiği yerine seyreltilmiş (sparsified proximitiy graph) yakınlık grafiği kullanılır. Bunun nedeni gürültünün ve aykırı gözlemlerin etkisini azaltmaktır. Seyrek bir grafik elde edildikten sonra, çok seviyeli garfik ayırıcı algoritmayla veri seti bölümlere ayrılır.

Chameleon veri setinin bölümlendirilmesi işlemini tüm kümeleri kapsayan en büyük kümeyi (all inclusive cluster) bölümlere ayırarak gerçekleştirir. Bu küme, tüm kümeler en az kullanıcı tarafından belirlenen bir parametre olan MIN-SIZE kadar noktaya sahip oluncaya kadar bölünür. Bu işlemle çok sayıda birbirine neredeyse eşit olacak kadar yakın büyüklükte ve içindeki noktalar birbirine çok benzeyen gruplar oluşturulur. Bu gruplar çoğunlukla gerçek bir kümeden noktalar içeren alt kümelerdir.

Alt kümeler, kendilerine benzeyen bir üst kümeyle hiyerarşik kümeleme algoritması kullanılarak birleştirilir. Kümelerin birleştirilmesinde iki farklı kriter dikkate alınır.

1. Göreli içbağlılık (Relative interconnectivity (RI)): İki alt küme, oluşturulacak yeni kümenin noktaları birbirine alt kümeler kadar güçlü bağlanmışsa birleştirilir.

$EC(C_i,C_j)$; C_i ve C_j kümelerini bağlayan k-en yakın komşu grafiğinin uç noktalarının toplamı,
$EC(C_i)$; eğer C_i' nin iki eşit parçaya bölünmesi halinde, kümenin kesilen kenarlarındaki noktaların minimum toplamı; ($EC(C_j)$ de benzer şekilde) olmak üzere, RI

$$RI = \frac{EC(C_i,C_j)}{\frac{1}{2}\left(EC(C_i)+EC(C_j)\right)} \tag{3.17}$$

şeklinde hesaplanır.

2. Göreli yakınlık (Relative closeness (RC)): İki küme, oluşturulacak yeni kümenin noktaları alt kümelerdeki noktalar kadar birbirine 'yakınsa' birleştirilir. RI'daki uç noktaların minimum toplamı yerine $\bar{S}$ ortalamaları alınır. RC aşağıdaki gibi hesaplanır.

$$RC = \frac{\bar{S}_{EC(C_i,C_j)}}{\frac{|C_i|}{|C_i|+|Cj|}\bar{S}_{EC(C_i)} + \frac{|C_j|}{|C_i|+|Cj|}\bar{S}_{EC(C_j)}} \tag{3.18}$$

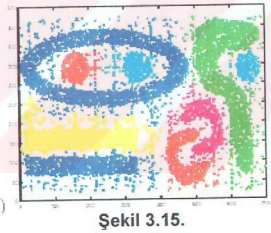
m nokta sayısı, p de küme sayısı olmak üzere, bu algoritmanın kapladığı alan $O(mp+m\log m+p^2\log p)$ olarak ifade edilir. Bu alan $O(m\log m)$ zaman içinde

k-en yakın komşu grafiğinin oluşturulmasına bağlıdır. Bu grafiğin çok büyük veri setlerinde oluşturulması oldukça fazla zaman gerektirir $\left(O(m^2)\right)$ ve algoritma büyük veri setleri üzerinde iyi çalışmaz.

Chameleon algoritması

1. k-en yakın komşu grafiği oluşturulur.
2. Bu grafik bir çok aşamalı grafik ayırıcı algoritmayla parçalara bölünür
3. Hiyerarşik bir algoritmayla bu bölümler küme benzerliği kavramına göre birleştirilir.

Aşağıda Şekil 3.9.' da kullanılan veri seti üzerinde Chameleon algoritması uygulanması sonucu oluşan kümeler gösterilmiştir. Chameleon'un parametreleri: k-NN=10, MinSize=%2.5 k=9 olarak alınmıştır.



Şekil 3.15.
Chameleon
(Zaiane, Foss, Lee and Wang, 2002)

3.3.3. Yoğunluğa dayalı kümeleme teknikleri

Yoğunluğa dayalı kümeleme teknikleri, yoğunluğa dayalı küme tanımından yola çıkılarak geliştirilmiştir. Bu teknik kapsamındaki algoritmalardan bazıları; CLIQUE, MAFIA (Merging of Adaptive Finite Intervals) ve DBSCAN' dir. Özellikle CLIQUE ve MAFIA çok boyutlu veri setleri üzerinde rahat kümeleme yapabilirler.

3.3.3.1. DBSCAN

DBSCAN kümeleme algoritması, veri setindeki her noktayı farklı uzaklık ölçülerine göre ya kümelere atar ya da bir gürültülü nokta olarak kabul eder.

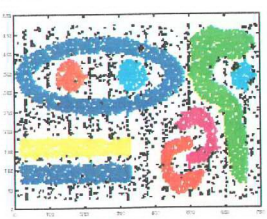
Çekirdek (core) nokta: Kümenin içinde olan noktalardır. Bir noktanın kümenin içinde sayılması için, çevresinde yeteri kadar nokta olması gerekir. Eğer iki çekirdek nokta birbirinin komşusu sayılıyorsa aynı kümeye dahildirler.

Sınır (border) nokta: Çekirdek nokta olmayan her nokta bir sınır noktadır. Bir sınır noktanın çevresinde yeteri kadar nokta yoktur fakat bir çekirdek noktanın komşusudur. Sınır nokta aynı anda başka kümelere dahil olan çekirdek noktaların komşusu olabilir. Böyle bir durumda, DBSCAN algoritmasının parametrelerine göre gruplama yapılır.

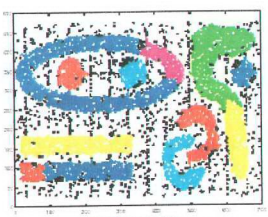
Gürültülü nokta: Çekirdek veya sınır nokta olmayan her nokta gürültülü nokta olarak kabul edilir.

DBSCAN algoritmasının parametreleri; bir noktanın başka bir noktaya komşu olması için hangi alan içinde bulunması gerektiğini belirten yarıçap (ε) ve bir kümenin sahip olabileceği minimum nokta sayısını belirleyen MinPts' dir.

Aşağıda Şekil 3.9.'da kullanılan veri setinin üzerinde faklı parametrelerle DBSCAN'ın



Şekil 3.16. DBSCAN
ε = 5.9 MinPts = 4



Şekil 3.17. DBSCAN
ε = 5.5 MinPts = 4

(Zaiane, Foss, Lee and Wang, 2002)

3.3.4. Grafik esaslı kümeleme teknikleri

Oluşturdukları kümeleri basit grafikler ya da hipergrafiklerle (hypergraph) gösteren kümeleme teknikleri grafik esaslı kümeleme teknikleridir. Bu kapsamdaki teknikler kullanılan grafiğin türüne göre ikiye ayrılırlar.

3.3.4.1. Grafiğe dayalı kümeleme teknikleri

Oluşturdukları kümeleri dendogramlarla gösteren hiyerarşik kümeleme teknikleri grafiğe dayalı kümeleme tekniklerdir. Bu teknikler en yakın komşu kavramından yola çıkarak kümeleme yaparlar.

En yakın komşu kavramı; bir noktanın en yakınındaki noktayla beraber aynı kümeye dahil olacağı prensibine dayanır.

Ortak en yakın komşu kümelemesinin (mutual nearest neighbour clustering) iki farklı tanımı vardır ve bu tanımlamalar farklı algoritmaların yaratılmasına neden olmuştur. Ortak en yakın komşu kümelemesinin ilk tanımına göre oluşturulan algoritmanın adımlar aşağıda açıklanmıştır.

1. Her nokta için en yakın komşusu bulunur.
2. Her noktanın kendisi ve komşusundan oluşan çift karşılaştırılır.

Karşılaştırmada şu iki nokta dikkate alınır ve 3 adıma geçilir.

a. Eğer iki nokta belli bir komşu sayısından daha fazla ve
b. İki nokta da birbirinin k-en yakın komşularına dahilse

3. Bu iki nokta ve bulundukları kümeler birleştirilir.

Farklı yoğunluktaki kümeleri ayırdedebilen bu kümeleme tekniğinin geçişli (transitive) bir yapısı vardır. Eğer p noktasının q noktasıyla ortak komşusu varsa ve aynı zamanda q noktasının r noktasıyla ortak komşusu varsa, p,q ve r noktaları aynı kümeye atanırlar. Böylelikle algoritma farklı boyutlarda ve şekillerdeki kümeleri de farkedebilir.

Ortak en yakın komşu kümelemesinin ikinci tanımı, noktaların en yakın komşularının sıralanarak sıra sayılarının toplanmasıyla elde edilen ortak komşuluk değerine (mutual neighbour value (mnv)) dayanır.

Ortak komşuluk değerlerinin birbirine yakın olduğu noktalar ortak en yakın komşu sayılırlar. Bu tanımdan yola çıkılarak oluşturulan algoritmanın adımları:

1. Her nokta için k-en yakın komşular bulunur.
2. Her noktanın bütün k-en yakın komşuları için nokta ve komşusu arasındaki ortak komşuluk değeri hesaplanır.
3. En düşük ortak komşuluk değerine sahip olan kümeler birleştirilir.
4. 3. adım istenilen kadar küme sayısı elde edilinceye kadar ya da kümeler artık birleştirilemeyinceye kadar tekrar edilir.

Bu tanıma göre yapılan kümeleme, farklı yoğunluk, büyüklük ve şekillerdeki kümeleri farkedebilir.

3.3.4.2. Hipergrafiğe dayalı kümeleme teknikleri

Hipergrafiğe dayalı kümeleme teknikleri, çok boyutlu uzay üzerinde kümeleme yapan tekniklerdir. Hipergrafiklerin, grafiklerden farklı herhangi bir köşesinin ikiden çok daha fazla ekseni birleştirmesidir.

3.3.4.3. ROCK (Robust Clustering using linKs)

Kategorik ve boolean tipli verileri kümelemek amacıyla geliştirilen ROCK, noktalar arasındaki uzaklığı; verilen bir eşik değerinden (θ) daha 'güçlü' olan ortak komşu sayısı olarak tanımlar ve noktaları ayırmada hiyerarşik kümeleme yaklaşımını kullanır.

ROCK sepet analizi verileri üzerinde kullanılır. Sepet analizinde kullanılan verilerin benzerliklerinin Öklit uzaklığıyla ölçülmesi uygun olmadığından benzerlik ölçütü olarak için Jaccard katsayısını kullanır.

Jaccard katsayısı, uzaklıkların hesaplanmasında yeterli bir ölçüttür fakat bazı durumlarda kümeler iyi ayrılmamış olabilir. Bundan dolayı noktalar arasındaki benzerlik için farklı bir ölçüt kullanılır.

p_i ve p_j noktaları arasındaki benzerlik $sim(p_i, p_j)$ ve $0 \leq \theta \leq 1$ olmak üzere

$$link(p_i,p_j) = \left|\{q : sim(p_i,q) \geq \theta\} \cap \{q : sim(p_j,q) \geq \theta\}\right| \quad (3.19)$$

denkleminde link (p_i p_j'); p_i ve $p_{j'}$ nin ortak komşularının sayısını verir. Eğer iki noktanın göreli olarak çok sayıda ortak komşusu varsa, bu iki nokta birbirine 'yakın' dır. Bu tanım birbirine yakın fakat farklı kümelere dahil olan sınır noktarda açığa çıkan sorunu gidermeye yöneliktir.

ROCK hatayı diğer kümeleme algoritmalarından daha farklı tanımlar:

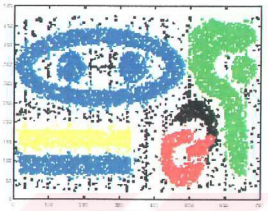
$$E = \sum_{i=1}^{k} n_i * \sum_{p_q,p_r \in C_i} \frac{link(p_q,p_r)}{n_i^{1+2f(\theta)}} \quad (3.20)$$

Amaç $link(p_q,p_r)$'nin mimimum olması olduğundan, algoritma E'yi maksimum kılacak şekilde tasarlanmıştır.

ROCK algoritması

1. Veri setinden örnek noktalar seçilir.
2. Tüm nokta çiftleri için Jaccard katsayısı kullanılarak link değerleri hesaplanır.
3. E' nin değeri maksimum olacak şekilde veriler üzerinde bir toplayıcı hiyerarşik kümeleme algoritması uygulunarak veri seti kümelere ayrılır.
4. Geri kalan noktalar kümelere atanır.

Aşağıda Şekil 3.9.'da kullanılan veri seti üzerinde ROCK'ın uygulanması sonucu oluşan kümeler gözükmektedir.



Şekil 3.18.
ROCK
(Zaiane, Foss, Lee and Wang, 2002)

3.4. Veri Madenciliğinde Kullanılan Kümeleme Tekniklerinin Özellikleri

Veri madenciliğinde sadece sayısal veriler üzerinde değil farklı nesneler üzerinde de kümeleme yapılır. Bir tür yapay sinir ağı olan SOM, veri madenciliğinde kümeleme yapmak için oldukça sık kullanılır. SOM' un algoritmasına yapay sinir ağları konusunda değinilmiştir.

Birçok kümeleme algoritması bazı varsayımların gerçekleşmesi durumunda optimal bir şekilde çalışabilmektedir. Varsayımların gerçekleşmemesi durumunda, kümeleri farkedemez, ayıramaz, birleştiremez ya da kümeler oluştururlar. Güçlü bir kümeleme algortimasının;

- Çok boyutlu
- Kayıp verili ve aykırı gözlemli
- Farklı dağılımlı
- Farklı şekilde kümelerin olduğu
- Farklı boyutlu kümelerin olduğu veri setleri
- Farklı yoğunluklu kümelerin olduğu

- Kümelerin çok iyi ayrılmamış olduğu
- Çok sayıda ve farklı türde değişkenlerin olduğu veri setlerinde kolaylıkla çalışabilmesi gerekir.

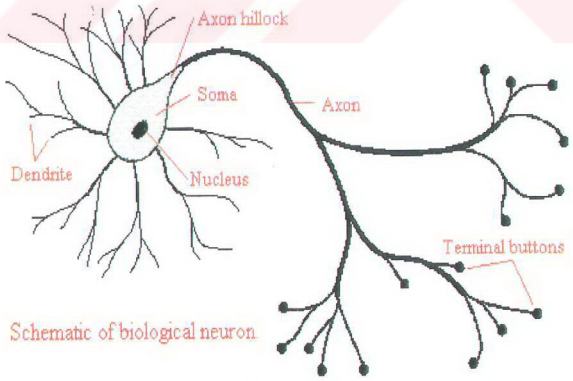
DÖRDÜNCÜ BÖLÜM
YAPAY SİNİR AĞLARI

Yapay sinir ağları, beyin hücreleri olan nöronların çalışma prensibini modelleyen öğrenebilen algoritmalardır. İleriye yönelik (feedforward) ya da geribesleme alabilen yinelemeli ağlar (recurrent) olmak üzere yapay sinir ağlarının iki türlü yapısı vardır.

Birçok yapay sinir ağı vardır fakat burada ele alınan, sınıflandırma ve kümeleme için kullanılan en temel sinir ağları olan Backpropagation algoritması, RBF ağları ve SOM' dir.

4.1. Biyolojik Sinir Ağları

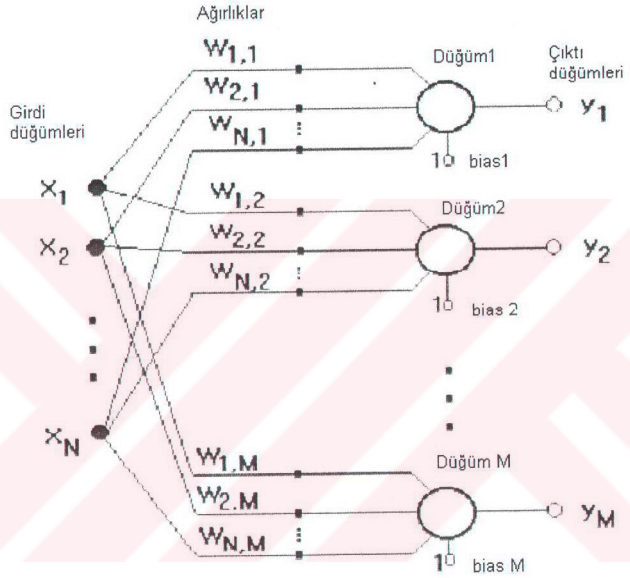
Aldığı sinyalleri aksonlar yardımıyla taşıyarak çekirdeğe ileten nöronlar birbirlerine dendritlerle bağlanırlar. Nöronlar arasında biyolojik olarak gerçekleşen sinir akışı, yapay sinir ağlarınca modellenmektedir.



Şekil 4.1.
Nöron

4.2. Yapay Sinir Ağları Bileşenleri

Amaçlarına ve işleyişlerine göre ayrılan birçok yapay sinir ağı türü vardır. Hepsinin ortak bileşenleri şu şekilde sıralanabilir:



Şekil 4.2.
Yapay sinir ağı

4.2.1. Yapay sinir hücreleri (artificial neuron)

Yapay sinir hücreleri (artificial neuron) aynı zamanda düğüm (node), birim (unit) ya da işlem elemanı (processing element) olarak da adlandırılır (Akpınar, 1993).

Her düğüm ağdaki diğer düğümlerden aldığı girdiyi (sinyal) başka düğümlere gönderir. Eğer girdi; düğüme başka bir düğümden değil de içinde bulunduğu çevreden geliyorsa, o düğüm girdi düğümü (input unit) adını alır. Benzer

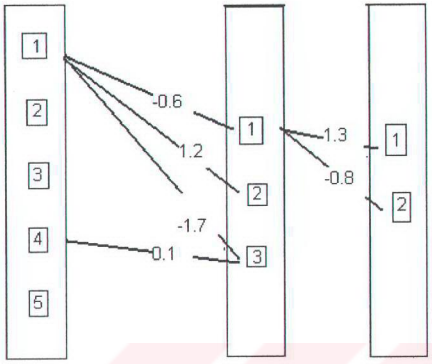
şekilde kendinden çıkardığı değeri başka düğümlere değil de çevreye gönderiyorsa, bu düğüm çıktı düğümüdür (output unit).

4.2.2. Düğümlerin bağlanma şekli

Düğümlerin birbirine bağlanış şekli ağın tipine göre değişir. Bir düğüm ağdaki tüm düğümlere bağlanabilir. Sıralı bir hiyerarşi izleyen katmanlarda bulunan düğümler, sadece ardışık katmanlardaki düğümlere; ya da ardışık katman içinde geriye dönük (feedback), kendi katmanı içindeki diğer düğümlerle veya kendisine bağlanabilir.

Düğümlerin birbirlerine bağlanması ağırlık anlamına gelir. Ağırlığı belirleyen 3 parametre vardır; bağlandığı düğüm, bağlanacağı düğüm ve ağırlığın değerini belirleyen skaler büyüklük. Ağılık değerinin negatif olması, düğümü engelleyici, pozitif olması düğümü destekleyici etki yaratır. Ağırlığın mutlak değeri ise gücünü belli eder.

i. düğümden j. düğüme ağırlık w_{ij} şeklinde gösterilir. Ağdaki tüm düğümlerin ağırlıklarını gösteren ağırlık matrisi, ağın işlemi nasıl yapacağına dair bilgiyi tutan hafızası olarak geçer. Şekil 4.3.'de 3 katmanlı bir yapay sinir ağı ve katmanlar arasındaki ağırlıkları gösteren ağırlık matrisleri verilmiştir.



Şekil 4.3.
3 katmanlı yapay sinir ağı

$$W_1 = \begin{bmatrix} -0.6 & 1.2 & -.17 \\ . & . & . \\ . & . & . \\ . & . & 0.1 \\ . & . & . \end{bmatrix} \qquad W_2 = \begin{bmatrix} 1.3 & -.08 \\ . & . \\ . & . \end{bmatrix}$$

Şekil 4.4.
Katmanlar için ağırlık matrisleri

4.2.3. Yayılım konrol kuralı

Bir ağırlığın bir düğümden diğerine taşınması sırasında düğümlerin nasıl güncelleneceği ve ne zaman diğer düğümlere gönderileceğini belirleyen kuraldır. Girdi ağılıklarını birleştirerek bir tek çıkış ağırlığı belirlemek, güncelleme yöntemlerinden biridir. Yapay sinir ağı tiplerine göre bu kural değişebilir. Bazı modeller güncellenecek düğümleri rasssal olarak seçerken, bazıları da bir grup düğüm güncellenmeden diğerlerinin güncellenmesine olanak tanımaz.

4.2.4. Girdi sinyallerini birleştirme kuralı

Bir düğüme gelen sinyallerin ağırlıklarla çarpımlarının toplamı en temel birleştirme kuralıdır. Başka bir birleştirme kuralı ise; düğümlere gelen sinyallerin ağırlıkları farkının karesinin toplamıdır. j düğümüne gelen ağırlıklar;

$$netj = \sum_{i=1}^{n} x_i w_{ij} \qquad (4.1)$$

şeklinde birleştirilir.

4.2.5. Çıktı değerini hesaplama kuralı

Bir düğümden diğer düğümlere ya da çevresine gönderdiği çıktı değerini hesaplama kuralı; aktivasyon fonsiyonu (activation function) olarak adlandırılır. Aktivasyon fonksiyonunun sonucu; aktivasyon fonksiyonuna bağlı olarak (0 ya da 1), (-1 ya da +1) ya da 0 ile 1 arasında bir sayı olabilir. Girdi düğümlerin aktivasyon fonsiyonu (I Identity fonksiyonu) özel bir ad alır. Aktivasyon fonksiyonları doğrusal, lojistik ya da hiperbolik fonksiyonlar olabilir. Doğrusal olmayan aktivasyon fonksiyonlarına sigmoid fonksiyonlar denilir.

4.2.6. Ağırlıkları gücelleme kuralı

Ağırlıklar düğümlere ulaştıktan sonra bir kısmı geriye dönebilir. Buna ağın hatası denir ve E = (pred-req) olarak formüle edilir.

Yapay sinir ağının E^{2} yi minimize edecek şekilde ağırlıkları uyarlaması gerekir. Backpropagation algoritması, azalan bir eğimle E^{2} yi minimize eden bir algoritmadır.

E^{2} yi mizimize etmek için öncelikle E^{2} nin ağırlıklara duyarlılığının hesaplanması gerekir. Başka bir deyişle, bir ağırlık değiştiğinde bunun E^{2} yi

nasıl değiştirdiğinin bilinmesi gerekir. Bunun bilinmesi halinde ağırlıklar E^2'yi azaltacak şekilde güncellenebilir.

Delta kuralı ya da Widrow-Hoff kuralı olarak geçer. Ağırlıklara göre hatanın değişiminin minimize edilmesine dayanır. A ve B iki düğüm olmak üzere; hatanın ağırlıklara göre değişimi;

$$\frac{\partial E^2}{\partial W_{AB}} \tag{4.2}$$

şeklinde ifade edilir. Bu değişim I identity fonksiyonu cinsinden yazılırsa;

$$\frac{\partial E^2}{\partial W_{AB}} = \frac{\partial E^2}{\partial I_B}\frac{\partial I_B}{\partial W_{AB}} \tag{4.3}$$

elde edilir.

$$\partial E^2 = \partial(\text{pred} - \text{req})^2 \tag{4.4}$$

(4.3)'deki $\frac{\partial E^2}{\partial I_B}$ ifadesi (4.4) ile gösterilen hata terimi cinsinden yazıldığında

ve eşitlik çözüldüğünde;

$$\frac{\partial E^2}{\partial I_B} = 2(\text{pred} - \text{req})\frac{\partial \text{pred}}{\partial I_B} \tag{4.5}$$

$$= 2E\frac{\partial f(I_B)}{\partial I_B} \tag{4.6}$$

$$= 2Ef'(I_B) \tag{4.7}$$

elde edilir.

B'nin çıktı düğümü olması durumunda; eğer çıktı aktivasyon fonksiyonu f(•) lojistik bir fonksiyonsa;

$$f(x) = \frac{1}{1+e^{-x}} = (1+e^{-x})^{-1} \tag{4.8}$$

bu fonksiyonun x' e göre türevi alındığında

$$f'(x) = -1(1+e^{-x})^{-2}.-1(e^{-x}) = \frac{e^{-x}}{(1+e^{-x})^2} \quad \text{elde edilir.} \tag{4.9}$$

$$f(x) = \frac{1}{1+e^{-x}} \tag{4.10}$$

Lojistik fonksiyondan e^{-x} çekilir;

$$f(x) = \frac{1}{1+e^{-x}} \Rightarrow e^{-x} = \frac{(1-f(x))}{f(x)} \tag{4.11}$$

ve (4.9)' un içine koyulursa

$$f'(x) = \frac{(1-f(x))}{f(x)} \Big/ \frac{1}{(f(x))^2} \tag{4.12}$$

elde edilir.

Benzer şekilde aynı işlemler hiperbolik fonksiyon için yapıldığında;

$$f'(x) = (1 - f(x)^2) \tag{4.13}$$

Lineer I fonksiyonu için yapıldığında,

$$f'(x) = 1 \text{ elde edilir.} \tag{4.14}$$

Sonuç olarak ; η öğrenme katsayısı parametresi olmak üzere

Lojistik fonksiyon için $\eta O_A 2EO_B(1 - O_B)$ (4.15)

Hiperbolik fonksiyon için $\eta O_A 2E(1 - O_B^2)$ (4.16)

Lineer fonksiyon için $\eta O_A 2E$ (4.17)

elde edilir.

B' nin bir gizli düğüm olması durumunda;

$$\frac{\partial E^2}{\partial I_B} = \frac{\partial E^2}{\partial I_o} \frac{\partial I_o}{\partial O_B} \frac{\partial O_B}{\partial I_B} \tag{4.18}$$

olacaktır.

$\frac{\partial O_B}{\partial I_B}$ aşağıdaki gibi yazılabilir.

$$\frac{\partial O_B}{\partial I_B} = \frac{\partial f(I_B)}{\partial I_B} = f'(I_B) \tag{4.19}$$

Tanım gereği $\frac{\partial I_o}{\partial O_B}$;

$$\frac{\partial I_o}{\partial O_B} = \frac{\partial \sum_p O_p W_{po}}{\partial O_B} \quad \text{ve} \tag{4.20}$$

$$\frac{\partial \sum_p O_p W_{po}}{\partial O_B} = \frac{\partial O_B W_{OB}}{\partial O_B} + \frac{\partial \sum_{p \neq B}^{p} O_p W_{po}}{\partial O_B} = W_{BO} \tag{4.21}$$

şeklinde yazılabilir. Yukarıdaki ifadeler kullanılarak (4.19)

$\frac{\partial E^2}{\partial I_B} = \frac{\partial E^2}{\partial I_o} W_{BO} f'(I_B)$ şeklinde yazılabilir. (4.22)

$\frac{\partial E^2}{\partial I_B}$; B'nin çıktı düğümü olması durumunda

$\frac{\partial E^2}{\partial I_B} = 2E f_o'(I_B)$ değerini (4.23)

gizli düğüm olaması durumunda

$\frac{\partial E^2}{\partial I_B} = \frac{\partial E^2}{\partial I_o} W_{BO} f_h'(I_B)$ değerini alır (4.24)

4.3. Sınıflandırma Amaçlı Yapay Sinir Ağları

Yapay sinir ağları sınıflandırma ve kümeleme amaçlı kullanılabilirler. Sınıflandırma amaçlı kullanılan yapay sinir ağları, delta kuralının açıklanmasında kullanılan backpropagation algoritması ve RBF ağlarıdır

4.3.1. RBF ağları (Radial Basis Function Netwoks)

Girdi, gizli (hidden) ve çıktı düğümlerinden oluşan üç katmanlı en basit yapay sinir ağıdır. Girdi düğümünden gizli düğüme doğrusal olmayan, gizli düğümden çıktı düğümüne ise doğrusal fonksiyon kullanır. Çoğunlukla gizli düğümlerin sayısı girdi düğümlerinden daha fazladır. Bunun nedeni aralarındaki fonksiyonun doğrusal olmamasıdır. j gizli bir düğüm olmak üzere; j düğümüne gelen sinyaller

$$net_j = \left\| x - w_{ij} \right\| \tag{4.25}$$

$$= \left[\sum_{i=1}^{n} \left(x_i - w_{ij} \right)^2 \right]^2 \tag{4.26}$$

şeklinde formülize edilir.

Backpropagation algoritmasının ve RBF ağlarının sınıflama problemleri için kullanılmaları durumunda sağladıkları avantajlar ve dezavantaljlar aşağıdaki gibidir;

Backpropagation algoritması:

- Hatayı minimize eden delta kuralına göre ağırlıkları güncelleyerek öğrenmeyi gerçekleştirir.
- Hata değeri yeteri kadar küçük olana kadar algoritma iteratif bir şekilde çalışır. Minimum hataya ulaşılsa bile, algoritmanın doğru sınıflandırma yapıp yapmadığı kontrol edilmelidir.
- Elde edilen ağırlıklar, farklı bir örüntüyü sınıflandırmak için kullanılabilir.
- Algoritmanın işelmesi uzun zaman alabilir.
- Gizli düğümlerin sayısı ve veri setinin büyüklüğü genelleştirmeyi etkiler.

RBF ağı

- Çözülecek problemi tüm yönleriyle temsil edecek yeterli sayıda veri gerektirir.
- Test edecek yeterli sayıda veri gerektirir. Test ve uygulama veri setleri ayrılmış olmalıdır.
- Gizli düğümler için aktivasyon fonksiyonunun önceden belirlenmiş olması gerekir. Genellikle Gaussien fonksiyon kullanılır.
- Gaussien fonksiyonların orta noktaları ağırlıkları için birinci katmanı oluşturur. Bu orta noktalar bir kümeleme tekniği ile belirlenebilir.

4.4. Kümeleme Amaçlı Yapay Sinir Ağları

80' lerin başında Kohonen tarafından geliştirilen Self-Organizing Feature Map, (SOM) kümeleme yapan bir yapay sinir ağıdır. SOM' de, girdi düğümleri uygulanan(training) veri vektörünün boyutlarını gösterirler, çıktı düğümleri ise temsili noktalara denk düşerler ve küme düğümleri olarak da adlandırılırlar.

SOM' de küme düğümlerinin ağırlıkları, kümelerin koordinat sistemi üzerindeki konumunu belirler.

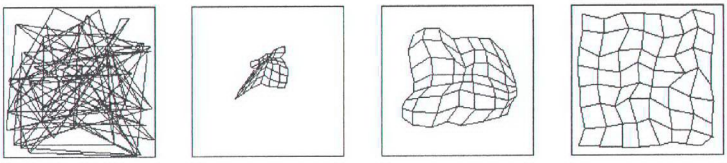
SOM' in algoritması

1. Ağırlıklara rassal sayılar atanır.
2. Her küme düğümü için uygulanan veri vektörüne olan uzaklık hesaplanır. Bu işlem bütün girdi vektörleri için yapılır.

$$d_j = \sum_i \left(w_{ij} - x_i \right)^2$$

3. j düğümü için mimimum uzaklık bulunur.
4. Kullanıcı tarafından belirlenen yarıçap içinde kalan tüm ağırlık vektöründeki düğümler, güncellenir.
5. Öğrenme oranı veya yarıçapın güncellenip güncellenmeyeceğine karar verilir.
6. 2. adıma geri dönülür.

Şekil 4.5.' de 1900 iterason sonrası SOM gösterilmektedir.



Başlangıç iterasyonu | 200. iterasyon | 600. iterasyon | 1900. iterasyon

Şekil 4. 5.
SOM

4.5. Yapay Sinir Ağlarının Özellikleri

Yapay sinir ağlarının özellikleri, aynı zamanda insan beynin de özelliklerini yansıtır:

- öğrenme ve genelleştirme kabiliyetleri
- uyarlanabilir olmaları
- içerik adresleyebilmeleri (content addressabilitiy)
- hata payını tolere edebilmeleri
- kendilerini organize edebilmeleri
- güçlü olmaları
- eşanlı işlemleri basitleştirmeleri

nedeniyle birçok alanda yapay sinir ağları tercih edilmektedir.

Yapay sinir ağları karar verme, hava tahmini ve mühendislik alanlarında (RBF ağı), kümeleme problemlerinde (SOM), fonksiyonel yaklaşımlarda, hafıza gerektiren problemlerde, pattern recognition alanlarında oldukça sık kullanılırlar.

4. 6. Yapay Sinir Ağları ve İstatistiksel Teknikler

Yapay sinir ağlarıyla istatistiksel teknikler arasında oldukça benzerlik vardır;

- Gizli düğümü olmayan ileriye yönelik ağlar, genelleştirilmiş lineer modellere,
- Bir tane gizli düğümü olan ileriye yönelik ağlar, projection pursuit regresyona,
- Olasılıksal yapay sinir ağları; kernel diskriminat analizine,
- Genel regresyon ağları, Nadaraga-Watson kernel regresyonuna,
- Hebbian öğrenim, temel bileşenler analizine benzer.

BEŞİNCİ BÖLÜM
GENETİK ALGORİTMALAR

Genetik algoritmalar, J. Holland tarafından (1975) ortaya atılan, yapay sinir ağları gibi biyolojik sistemlerden esinlenerek tasarlanmış bir tür makine öğrenimi tekniğidir. Evrim teorisini modelleyen bu algoritmalar, doğal şartlara en fazla uyum sağlayabilen bireyin hayatta kalabilmesi (survival of the fittest) temeline dayanan doğal seleksiyonu bilgisayar üzerinde simüle ederler.

5.1. Biyolojik Kavramlara ve Genetik Algoritmalara İlişkin Tanımlamalar

Bir populasyon, en basit anlamda birbiriyle etkileşim içinde olan bireylerden oluşan bir topluluktur. Populasyon içinde jenerasyonlar sürekli olarak yenilenirler. Bu yenilenme bir jenerasyondaki güçlü, çevresine uyum sağlayabilen bireylerin üreyerek özelliklerini yeni bireylere aktarmalarıyla gerçekleşir.

Üreme, genetik bilgiyi taşıyan genomların ebeveynlerden çocuklara (offspring) geçmesidir. Moleküler düzeyde ise bir çift kromozomun genetik bilgilerini crossover ile birbirine aktarmasıyla gerçekleşir. Üreme sürecinde herhangi bir bireyde oluşabilecek mutasyon, ait olduğu populasyondaki çeşitliliğin artmasına yol açar. Evrim teorisinin çeşitli türlerden oluşan bir populasyonun bireylerinin çevre koşullarına uyum sağlayacak şekilde adaptasyon geçirmesine olanak sağlaması, genetik algoritmalar için de geçerlidir.

Biyolojik sistemlerin en belirgin özellikleri güçlü (robustness) ve esnek (flexibility) olmalarıdır. Genetik algoritmaların güçlü, esnek ve geniş problem uzaylarında etkili oldukları Holland tarafından ispatlanmıştır.

Genetik algoritmalarda crossover, çevresine uyum sağlayabilen bireylerin 'seçilebileceği' ortamlarda gerçekleşir. Çevreye uyumun bir ölçüt, çevreye

uyum sağlayan bireylerin seçiminin de bir fonksiyon olduğu düşünülebilinir. Bazı genetik algoritmalar, uyum sağlayan bireylerin seçimini doğada olduğu gibi (crossover ya da eşeysiz üreme) olasılıksal bir şekilde gerçekleştirir. Bu tür seçime uyum-orantılı seçim (fitness-proportionate selection) denir. Başka bir yöntem de populasyonun bir alt grubundan rassal olarak seçilen bireylerden en iyi uyum sağlayanının seçimidir. Bu tür seçime turnuva seçimi (tournament selection) denir. Kullanılan diğer seçim teknikleri; bireylerin uyum değerlerine doğrusal ve üssel olarak ağırlandırılarak sıralanmasına dayanan sıralı seçim teknikleri (Linear ve Exponential Ranking Selection) ile orantısal seçim tekniği olan Roulette Wheel Selection' dır. Evrime en fazla katkıyı sağlayanlar crossover ve uyuma dayanan seçim yöntemleridir.

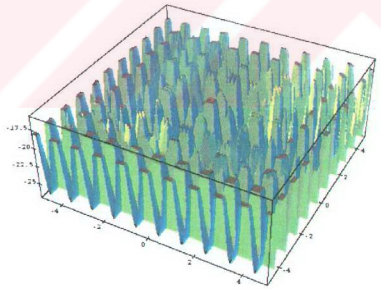
Genetik algoritmalar, bilgisayar ortamında rassal olarak bir populasyon üretirler. Bu populasyonun bireyleri, önceden belirlenen uzunlukta (bit olarak) string ifadelerdir. String ifadeler 0, 1 ve * karakterlerinden oluşur. Bireyler algoritmanın türüne göre farklı fonksiyonlarla üretilerek çoğaltılırlar. Uyum fonksiyonundan (fitness function) elde edilen uyum değerine (fitness value) göre bireylerin hangilerinin hayatta kalacağı belirlenir. Oluşan yeni jenerasyon istenilen sonucu vermiyorsa optimal bir sonuç elde edilene kadar bireyler üretilmeye devam edilir. Üretme sürecinde bireyler oldukça düşük bir olasılık kullanılarak mutasyona uğratılırlar. Bu da doğada olduğu gibi farklı türlerin ortaya çıkmasına neden olur. Başlangıç populasyonundan farklı türde string bireylerin oluşması, algoritmayı rassal yürüyüşe sürükleyecektir.

Genetik algoritmaların farklı uygulama alanları vardır. Ekonomide oyunlar teorisinde, çok boyutlu optimizasyon problemlerinde, fonksiyon optimizasyonunda, yapay sinir ağlarının oluşturulmasında ve denenmesinde, biyolojide protein yapılarının belirlenmesinde, veritabanlarında sorgu optimizasyonunda, programlama problemlerinde (gezgin satıcı problemi), suçlu teşhisinde, depremlerin merkez üslerinin belirlenmesinde, uçak tasarımında ve beste yapımında kullanılırlar.

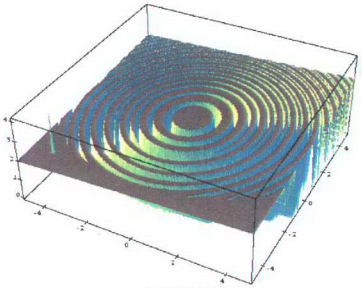
Çok boyutlu optimizasyon problemlerinde optimize edilmek istenen farklı parametreler ilk populasyonun bireyleri olarak kabul edilirler. İlk ebeveynlerden üretilen jenerasyonlar kullanıcı tarafından belirlenen kısıtlara göre optimal sonuç elde edilinceye kadar üretilirler.

5.2. Arama Teknikleri

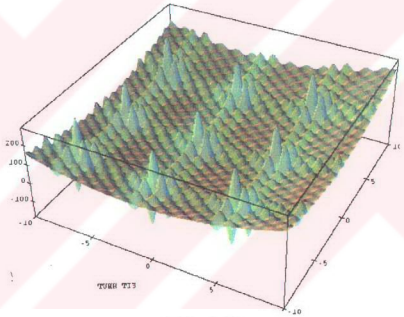
Genetik algoritmalar optimizasyon için kullanıldıklarında, belirli bir tür problem uzayı üzerinde optimum noktayı 'arayan' arama tekniklerine dönüşürler. Arama yaparken rassallık kavramını kullanan bu teknikler gerçek anlamda rassal arama algoritmaları değillerdir. Problem uzaylarına Hills, Plateaus, More Plateaus, Bryce Canyon, Rippled ve Levy's Egg Cartoon örnek olarak verilebilir. Problem uzaylarından Plateaus, Rippled ve Levy's Egg Cartoon Şekil 5.1, 5.2 ve 5.3'de gösterilmiştir (www.cs.qub.ac.uk/~M.Sullivan, 2002).



Şekil 5.1.
Plateaus Problem Uzayı



Şekil 5.2.
Rippled Problem Uzayı



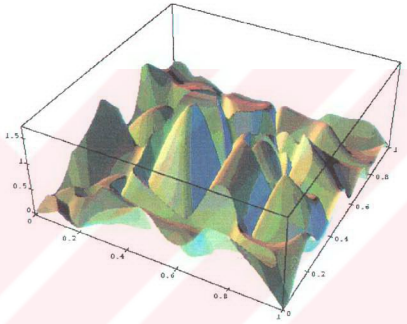
Şekil 5.3.
Levy's Egg Cartoon Problem Uzayı

5.2.1. Hill Climbing teknikleri

Hill Climbing teknikleri, 18. ve 19. yüzyılda klasik matematikçiler tarafından geliştirilmiş olan optimizasyon teknikleridir. Bu teknikler bir fonksiyonun lokal eğimini izleyerek bir optimum nokta bulmayı amaçlarlar. Eğim teknikleri olarak da adlandırılırlar. Deterministiktirler.

Hill Climbing tekniklerinin dezavantajı, üzerinde arama yaptıkları problem uzayının sürekli olduğu varsayımına dayanmalarıdır. Bu varsayım, problem uzayını yansıtan fonksiyonun türevi olmasını gerektirir. Oysa gerçek yaşamda problem uzayları daha çok kesiklidir ve kayıp veriler çoktur. Hill

Climbing tekniklerinin diğer bir dezavantajı ise, bir noktaya en yakın lokal minimumu bulmalarıdır. Bu yüzden genel bir bakışa olanak tanımazlar. Paralel Hill Climbing teknikleri, problem uzayında bir çok noktayı aramak için kullanılırlar. Fakat bu teknikler de optimum bir değer bulmayı, özellikle farklı yükseklikte lokal tepelerin ve gürültülü verilerin olduğu uzaylar üzerinde garantilemezler



Şekil 5.4.
Hills Problem Uzayı

5.2.2. Sıralı arama algoritmaları

Sıralı arama algoritmaları, kesikli problem uzayında optimum değeri bulmak için fonksiyonun uzaydaki her noktasını araştırır ve değerlendirir. Özellikle büyük problem uzaylarında çok hantal kalan algoritmalardır.

5.2.3. Rassal arama algoritmaları

Rassal arama algoritmaları, o ana kadar elde edilen optimum değerleri kaydederek problem uzayında rassal yürüyüşle arama yapar. Bu algoritmalar büyük problem uzaylarında sıralı arama algoritmalarından daha iyi sonuç vermezler.

Rassal arama algoritmaları daha önce elde edilen sonuçlardan elde edilen bilgiyi kullanamazlar. Bu yüzden yönsüz ve kör algoritmalar olarak değerlendirilirler

5.2.4. Rassallaştırılmış arama algoritmaları

Rassallaştırılmış arama algoritmaları, problem uzayını ararken rassal olarak ilerleyen basit rassal arama algoritmaları gibi 'yönsüz' değillerdir. Problem uzayını ararken rassal olarak ilerleyen basit birer rassal yürüyüş değillerdir. Daha önceki arama sonuçlarından sağladıkları bilgiyi, rassal olarak elde ettikleri bilgiyle birleştirirler. Gürültülü, çokboyutlu uzaylarda iyi sonuç verirler. Rassallaştırılmış arama algoritmalarından en önemlileri genetik algoritmalar ve simulated annealig' dir.

5.3. Geleneksel Arama Algoritmaları ile Genetik Algoritmaların Karşılaştırması

1. Genetik algoritmalar fonksiyonların gerçek parametreleriyle değil, string olarak kodlanmış şekilleriyle çalışırlar.
2. Aramaya başlamak için problem uzayında tek bir noktayı değil, bir nokta populasyonunu ele alırlar.
3. Problem uzayında arama yaparken önceki aramalardan elde ettikleri bilgiyi kendilerini yönlendirmek için kullanırlar.
4. Geleneksel arama algoritmaları gibi deterministik değil, olasılıksaldır.
5. Genetik algoritmalar paraleldirler. Eşanlı olarak birden fazla noktayla arama yapabilirler.

5.4. Genetik Algoritmaların İşleyişi

Genetik algoritmaların işleyişini açıklamaya başlamadan önce konu ile ilgili bazı temel kavramların tanımlanması uygun olur.

Genetik algoritmalar string bireylerden oluşan populasyonlar üzerinde çalışırlar. String bireyler 0,1 ve * karakteriyle ifade edilirler. Bu karakterler bireylerin genetik bilgilerine karşılık gelirler.

Biyolojik olarak ebeveynlerden çocuklara geçen genetik bilgi, genetik algoritmalarda tablolar içinde, birer 'şema' olarak ifade edilir. Bu tablolar schemata ya da building blocks ismini alır. Böyle bir tablo örneği aşağıda gösterilmiştir.

Tablo 5.1.
Building Blocks (Schemata)

Şema	Uyan Stringler
*1111	01111
	11111
*010*0	101010
	001010
	101000
	001000
**1*0	00100
	.....

Şemalarda kullanılan * karakteri, (önemsememe terimi) üremeden sonra kendisinin yerine 0 ya da 1 karakterinin gelebileceğini gösterir. **1*0 şeması, içerdiği * karakter sayısından dolayı *1111 şemasına göre daha fazla sayıda alt string (substring) oluşturur. Şemalara göre üretilen stringlerden uyum fonksiyonuna bağlı olarak 'yaşamlarına devam edebilenler' bir sonraki jenerasyonu oluşturmak için eşleşme havuzuna (mating pool) gönderilirler.

Crossover'ın building blocks üzerinde etkisi büyüktür. Crossover ile uyum sağlayan şemalar değişebilieceğinden crossover her aşamada uygulanmaz. Crossover'ın uygulanıp uygulanmayacağı algortima tarafından kararlaştırılır.

Bir şemanın uzunluğu, (ℓ) ve sırası crossover için gerekli olan bilgilerdir. Tablo 5.2.' de gösterildiği gibi şemanın uzunluğu; şemadaki belli olan (0 ve/veya 1) ilk ve son string elemanın arasındaki uzaklıktır.

Tablo 5.2.
Şemanın uzunlukları

Şema	Uzunluk (ℓ)
0*	0
1**1**	3
*0*1*1	4

Bir şemanın sırası ise, şemadaki belli olan (0 ve 1) string sayısıdır.

Tablo 5.3.
Şemanın sıraları

Şema	Sıra
*0***	1
10*0**	3
111**0	4

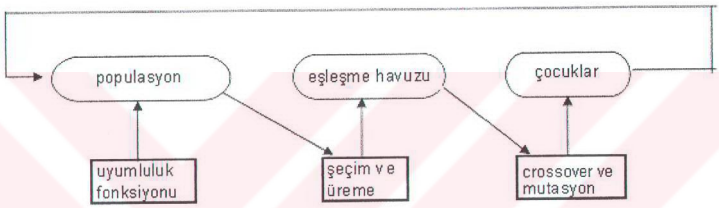
Bir şemanın sırası arttıkça şema daha spesifik hale gelir. Crossover' daki küçük sıralı şemaların büyük sıralılara göre bozulma şansları daha azdır. Kısa ve uyum gösteren şemaların, yeni jenerasyonu üretme olasılığı daha fazladır.

Genetik algoritmaların işleyişi

1. Problem uygun string olarak kodlanır.
2. Sonuçların çatışabileceği sanal bir ortam yaratılır. Başarı ya da başarısızlık için bir ölçüt belirlenir. (uyum fonksiyonu)
3. Sonuçların birleşebileceği bir yöntem hazırlanır. Çoğu algoritmada üreme için crossover kullanılır. Üreme esnasında oluşan tüm olası mutasyonlar gerçekleşebilir olmalıdır.
4. İyi bir başlangıç populasyonu oluşturulur. Her jenerasyon sonucu oluşan kötü sonuçları, mutasyon gibi bir yöntemle iyi sonuçlarla değiştiren bilgisayar üzerinde sanal bir evrimin geçekleşeceği ortam yaratılır.
5. Başarılı sonuçlardan oluşan bir aileyle karşılaşıldığında durulur.

5.5. Genetik Algoritmaların Operasyonları

Genetik algoritmalar, işleyişleri sırasında üreme, crossover ve mutasyon olmak üzere 3 farklı operasyon gerçekleştirirler. Bu operasyonlar en temel olanlardır. Genetik algoritma karmaşıklaştıkça, kullandıkları operasyonlar da zorlaşır. Belli başlı yüksek seviyeli operasyonlar; Dominance & Diploidy, Inversion & Reordering, Niche & Speciation, The Islands Model ve Spatial Mating olarak sayılabilir (www.cs.qub.ac.uk/~M.Sullivan, 2002).



Şekil 5.5.
Temel genetik algoritma operasyonları

5.5.1. Üreme

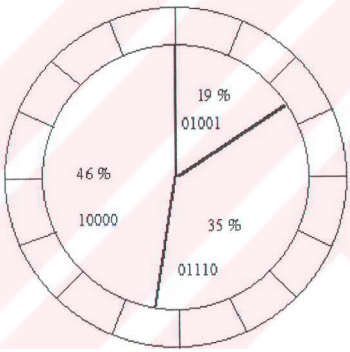
Şemaya ait bir string bireyin üremesi, uyumluluk fonksiyonundan elde edilen uyumluluk değerine bağlıdır. Genetik algoritma, bir sonraki jenerasyonu her jenerasyonda eşleşebilecek stringleri seçerek ve eşleşme havuzuna atarak oluşturur. Farklı genetik algoritmalar, farklı üreme yöntemleri kullanırlar.

Üreme yöntemlerinden en yaygın kullanılanı, Roulette Wheel Selection yöntemidir. Bu yöntemde, önce eşleşecek stringler seçilirken, stringlerin uyumluluk değerleri bağıl olarak orantılanır.

Tablo 5.4.
Stringlerin Uyumluluk değerleri ve yüzdeleri

String	Uyumluluk Değeri	Yüzde
01001	5	19%
10000	12	46%
01110	9	35%

Daha sonra stringler orantılarıyla orantılı yer kaplayacak şekilde bir ruletin üzerine yerleştirir. Rulet her defasında 3 kere döndürülerek eşleşme havuzuna atılacak stringin seçimi gerçekleştirilir. En yüksek orantılı stringin eşleşme havuzuna atılma olasılığı diğerlerinden daha yüksektir. Eşleşme havuzuna atılan string rulet üzerinden çıkartılmaz, aynı string bir kereden fazla seçilebilir. Böylelikle eşleşme havuzunda 'güçlü' stringler doğada olduğu gibi dominant olurlar. Aşağıda Tablo 5.4.' de bulunan stringlerin, uyumluluk değerlerinin yüzdelerine göre rulet üzerine yerleştirilmesi gösterilmiştir.



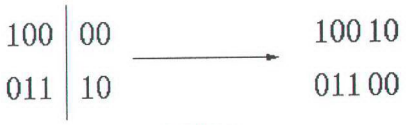
Şekil 5.6.
Roulette Wheel Selection

5.5.2. Crossover

Eşleşme havuzu yaratıldıktan sonra algoritma rassal olarak havuzdan 2 tane string seçer ve kullanıcı tarafından belirlenen bir parametre olan p_c (crossover olasılığı) değerine göre crossover'ın gerçekleşip gerçekleşmeyeceğine karar verir. p_c değeri genellikle 0.6 olarak alınır.

Eğer genetik algoritma crossover' ın gerçekleşmemesine karar verirse, eşleşecek 2 string olduğu gibi yeni jenerasyona aktarılır ve eşleşme havuzundan çıkartılmaz. Crossover'ın gerçekleşmesi kararı verilirse,

algoritma tarafından rassal olarak bir bölümlendirme noktası belirlenir. Stringler bu bitten itibaren bilgilerini değiştirir. Crossover'ın nasıl çalıştığı Şekil 5.7.' de özetlenmiştir.



Şekil 5.7.
Crossover'ın rassal olarak seçilen
4. bit üzerinde gerçekleşmesi

Yeni jenerasyon tamamlandıktan sonra, döngü seçim ile yeniden başlar. Kullanıcı tarafından belirlenen bir durma kriterine kadar (belli bir sayıda jenerasyon ürettikten sonra ya da bir eşik değerini geçen string elde edilinceye kadar) tekrarlı süreç devam eder.

5.5.3. Mutasyon

Seçim ve crossover büyük sayıda ve farklı stringler üretmeye yeterlidir. Fakat başlangıç populasyonuna bağlı olarak, yeteri kadar türde string olmaması durumunda; genetik algoritma problem uzayını bir bütün olarak göremez. Bu sorunu önlemek için stringler algoritma tarafından mutasyona uğratılır.

Kullanıcı tarafından belirlenen m parametresi (mutasyon olasılığı) mutasyonun hangi sıklıkla gerçekleşeceğini belirler. Bu olasılık 0,001 gibi oldukça düşük tutulur, çünkü yüksek mutasyon olasılığı, uyum sağlayan stringlerin dejenere olmasına neden olur. Bu da genetik algoritmayı rassal yürüyüşe sürükler.

Genetik algoritma, eşleşme havuzundaki bütün stringlerin her bitini mutasyona uğratıp uğratmayacağına karar verir. Mutasyon söz konusu olacaksa değiştirilecek olan bitin yerine rassal olarak başka bir değer atar.

Şekil 5.8.'de bir mutasyon örneği gösterilmektedir. Stringe ait 4. bit rassal olarak mutasyona uğratılmıştır.

$$10000 \xrightarrow{\text{Mutasyon}} 10010$$

şekil 5.8.

Mutasyon, seçim ya da crossover zamanında yapılabilirse de çoğunlukla tercih edilen crossover aşamasıdır.

5.6. Matematik Karşılıklar

ℓ uzunluğundaki bir string, 2^{ℓ} farklı şekilde oluşturulabilir. ℓ uzunluğundaki stringden $2^{2\ell}$ tane alt string üretilir. * karakterinden dolayı 3^{ℓ} tane şema oluşturulur.

Beklenen değer

H; belirli bir populasyonda, t zamanında en az bir tane örneğe (instance) sahip olan bir şema,

$m(H,t)$; H şemasının t zamanındaki örnek sayısını belli eden fonksiyon;

$\hat{u}(H,t)$; H şemasının t zamanındaki gözlenen ortalaması olsun.

Amaç, bir sonraki zaman (t+1) için H şemasının beklenen örnek sayısını hesaplamaktır. Yani $E(x)$, x'in beklenen değeri olmak üzere, $E(m(H,t+1))$ hesaplanmalıdır.

(t+1) zamanında, örneklerin beklenen değeri üretilen çocuk sayısına eşittir. Bu sayının bulunması, kullanılan seçim yöntemine göre değişir. Şeçim yönteminin uyum-orantılı seçim olması durumunda:

Uyum oranı; bir stringin uyum değerinin, jenerasyon ortalama uyum değerine oranı olur aşağıdaki gibi formüllenir.

$$\text{uyum\_orantısı} = \frac{f(x)}{\bar{f}(t)} \tag{5.1}$$

x, H şemasının bir örneği olmak üzere hesaplanmak istenen, t+1 zamanında H şemasının beklenen değeridir. Buna göre (5.1)' le bağlantılı olarak,

$$E(m(H,t+1)) = \sum_{x \in H} \frac{f(x)}{\bar{f}(t)} \tag{5.2}$$

elde edilir. $\hat{u}(H,t)$ tanımı gereği aşağıdaki gibi yazılabilir.

$$\sum_{x \in H} \frac{f(x)}{m(H,t)} = \hat{u}(H,t) \tag{5.3}$$

(5.2) ve (5.3) birleştirilerek sonuç olarak

$$E(m(H,t+1)) = \frac{\hat{u}(H,t)}{\bar{f}(t)} m(H,t) \qquad \text{elde edilir.} \tag{5.4}$$

Hayatta kalma

H şemasının bir örneğinin ebeveyn olarak ele alındığı varsayılsın. Üreme sonrası oluşan çocuklardan herhangi biri H şemasının örneklerinden biriyse, H şemasının hayatta kalmayı başardığı kabul edilir. Örneğin "1 1 * * * * " şemasının üretilmesiyle elde edilen 1100010 ve 010100 string çocukları dikkate alalım. Bu iki çocuktan ilkinin şemanın örneği, fakat ikincisinin ise şemanın örneği olmadığı görülebilir.

Çocukların 4. bitleri üzerinde crossover uygulandığı varsayılsın ve üretilen çocuklar 110000 ve 010110 olsun. Birinci string hala şemanın örneği olduğundan H şeması hayatta kalmıştır.

p_c; populasyon içinde belli bir stringe crossover uygulanma olasılığı,

ℓ ; üretilebilecek stringlerin bit olarak uzunluğu,

$d(H)$; tanımlanan uzunluk olmak üzere;

H şemasının crossover uygulandıktan sonra hayatta kalma olasılığı $S_c(H)$ için aşağıdaki eşitsizlik geçerlidir;

$$S_c(H) \geq 1 - p_c \frac{d(H)}{\ell - 1} \qquad (5.5)$$

Bu ifadede $\frac{d(H)}{\ell - 1}$ oranı; şemanın sahip olduğu string oranını verir. 5 bit uzunluğunda bir sitringin 3 biti tanımlanıyorsa oran 3/(5-4)= %75 olur.

Bu oranın p_c olasılığıyla çarpımı, şemanın yıkıma uğrayacak kısmı için üst sınırdır. Alt sınırı elde etmek için bu değerin 1' den çıkartılması gerekir.

Mutasyon

Mutasyondan sonra şemanın hayatta kalabilmesi için, mutasyonun etkisiz olması, yani şemanın değişmemesi gerekir. Bunun için şemanın mutasyondan sonra hayatta kalma olasılığı, bir bitin mutasyona uğramama olasılığının şema sırası kadar üssüne eşit olması gerekir.

p_m ; 1 bitin mutasyona uğrama olasılığı,

o(H) şemanın sırası olmak üzere;

$S_m(H)$; H şemasının mutasyondan sonra hayatta kalma olasılığı;

$S_m(H) = (1 - p_m)^{o(H)}$ olarak ifade edilir. (5.6)

Kısa ve düşük sıralı şemaların hayatta kalma şansları daha fazladır.

(5.5) ve(5.6) ifadelerinin (5.4) ifadesine eklenmesiyle,

$$E(m(H,t+1)) \geq \frac{\hat{u}(H,t)}{\bar{f}(t)} m(H,t)(1-p_c \frac{d(H)}{\ell-1})\left((1-p_m)^{o(H)}\right) \quad (5.7)$$

elde edilir.

$p_i(t)$; t zamanında populasyon içindeki i stringinin oranı,

s(t); stringin eşleşme olasılığı,

F_{ij}: uyum fonksiyonu olmak üzere; (Uyum fonksiyonu 2 boyutlu bir matristir. i=j hariç diğer bütün durumlar için 0' a eşittir.)

$$\bar{s}(t) = \frac{F\bar{p}(t)}{\sum_{j=0}^{2^{\ell}-1} F_{ij} p_j(t)} \quad (5.8)$$

ifadesi eşleşme olasılığını verir.

Algoritmanın jenerasyon üzerinde çalışması;

$$\bar{s}(t+1) = G\bar{s}(t) \quad (5.9)$$

ile ifade edilebilir. t zamanındaki eşleşen stringler üzerinde algoritmanın çalışması, t+1 zamanındaki eşleşecek olan stringleri verir.

$$E(\bar{p}(t+1)) = \bar{s}(t) \quad (5.10)$$

İfadesi, t+1 zamanındaki populasyonun beklenen değeri, t zamanında stringlerin eşleşme olasılığına eşit olduğunu gösterir.

(5.9)'un uyum fonksiyonu cinsinden ifadesi aşağıda verilmiştir.

$$\bar{s}(t+1) \square F\bar{p}(t+1) \qquad (5.11)$$

(t+1) zamanında eşleşecek stringlerin beklenen değeri, t zamanında uyum göstererek eşleşen stringlere denktir.

$$E(\bar{s}(t+1)) \square F\bar{s}(t) \qquad (5.12)$$

5.7. Genetik Algoritmaların Yapay Sinir Ağlarında Kullanımı

Genetik algoritmaların yapay sinir ağlarında kullanımı 3 ana başlıkta toplanabilir:

1. Ağırlıkların düzenlenmesinde denetimsiz ve destekli öğrenim uygulaması olarak ve farklı öğrenen algoritmaların kullanımı için öğrenme oranının belirlenmesinde kullanılır.
2. Genetik algoritmalar ağın yapısını öğrenmek için kullanılırlar. Ayrıca yapay sinir ağları fonksiyonel yaklaşım için kullanıldığında, yapay sinir ağında kaç tane gizli düğümü olması gerektiği ve düğümlerin nasıl bağlanması gerektiğini belirlemede genetik algoritmalar kullanılırlar.
3. Yapay sinir ağlarına uygun olan (training) verileri seçmede ve ağ çıktısınının açıklanmasında kullanılırlar (Craven and Shavlik, 1997).

ALTINCI BÖLÜM
GÖRSEL VERİ MADENCİLİĞİ

Yakın geçmişte yaygın biçimde kullanılan veri tiplerinin yanısıra, günümüzde artık daha çok yazılı metinler, 3 boyutlu nesneler, haritalar, fotoğraflar, resim ve ses dosyaları da dahil olmak üzere birçok farklı veri tipinin saklandığı veritabanları kullanılmaktadır. Bu veritabanları üzerinde yapılan analizler, analitik yöntemlerin haricinde hızlı ve kolay algılanabilir olduklarından görselleştirmeye yönelik teknikleri de kapsamaktadır.

Veri madenciliğinin daha etkin olabilmesi için, veri setlerinde uygulanan istatistik, makine öğrenimi, yapay zeka algoritmalarını desteklemek amacıyla verilerin görsel olarak da araştırılması neredeyse zorunluluk olmuştur.

W. S. Cleveland; veri setine 'en iyi uyan' tek bir grafik olmadığını ileri sürerek görselleştirmeyi veri dönüşümü, değişken indirgeme, düzleştirme gibi teknikleri içeren iteratif ve farklı açılardan farklı tür grafiklerle yaklaşılması gereken keşifsel (exploratory) bir süreç olarak tanımlamıştır (Mcleod and Provost, 2001). Görsel veri keşfi (visual data exploration) kavramı, bu noktada ortaya atılmıştır.

Özellikle veri seti hakkında çok az bilginin olduğu ve amacın tam olarak açıklanamadığı durumlarda önem kazanan görsel veri keşfi, farklı disiplinlerin kendi araştırma amaçlarına göre geliştirdiği, verilerin uzaydaki yer, boyut, şekil, yön, renk, ton ve dokularını ekran üzerinde nokta, doğru, alan veya hacim olarak görüntülenmesini sağlayan görselleştirme teknikleri (visualizing techniques) olarak adlandırılan algoritmaların büyük veri setleri üzerinde uygulandığı bir süreçtir (Niggeman, 2001).

Veri seti üzerinde arananın 'bilinmediği' durumlarda kullanıcıya hipotez kurmada yardımcı olmaları, istatistik, makine öğrenimi gibi disiplinlerin

kullandığı analitik algoritmaların ortaya çıkardığı sonuçları görsel olarak desteklemeleri ve kullanılmaları durumunda verilerin daha hızlı ve kolay yorumlanabilir olması, gösterim tekniklerine ilgiyi arttırmış ve daha sık gereksinim duyulmasına neden olmuştur. Özellikle analitik tekniklere göre homojen olmayan ve kayıp verili büyük veri setlerinde avantaj sağlamaları, daha hızlı sonuç çıkartmaları ve analitik algoritmaların çalışmadığı özel durumlarda çok daha iyi sonuç vermeleri nedeniyle, sadece belli bir tip görselleştirme tekniğini uygulayan programlar yazılmakta, veri madenciliği programlarına gösterim tekniklerini uygulayan araçlar yavaş yavaş ilave edilmektedir.

6.1. Tarihçe

İstatistikte özellikle Tukey'in geliştirdiği 2 ve 3 boyutlu statik gösterim teknikleri halen kullanılmaktadır. Fakat bilgisayarın günlük hayata girmesinden sonra veri setlerinin büyümesi, görselleştirme tekniklerinin daha hızlı ve kolay uygulanabilir hale gelmesi, özellikle istatistiksel gösterim tekniklerinin büyük veri setleri üzerinde uygulanması durumunda güçlerinin azaldığının saptanması, daha farklı görselleştirme tekniklerinin geliştirilmesine neden olmuştur.

İstatistikte aykırı gözlemlerin saptanması, grup veya kümelerin belirlenmesi, boyut indirgeme gibi amaçlar için kullanılan gösterim teknikleri, büyük veri setlerinin görselleştirilmesinde yetersiz kalırlar. Örnek olarak;

- Veri setindeki özel durumları ve boşlukları gösteren nokta grafikler (dotplot) 100 veriye kadar uygun görselleştirme sağlar fakat büyük veri setlerinde boşluklara nadiren rastlanır ve nokta grafiklerin özel durumları göstermeleri neredeyse imkansız bir hal alır. Benzer şekilde;
- aykırı gözlemleri gösteren boxplot' lar 100 veri içerisinde 2 - 3 tane aykırı gözlemi ideal olarak gösterebilirler, fakat milyonlarca veriden oluşan bir veri setinde aykırı gözlemlerin sayısı çok fazla olacağından özel gösterimlerin bir anlamı kalmayacaktır.

- Veri setinin dağılımı hakkında bilgi veren fakat aykırı gözlemleri göstermeyen histogramlar, büyük veri setlerine uygulandığında diğerlerine göre daha az gözlem içeren histogram aralıkları grafik alanda gösterilemez olur.
- Kategorik verilerin gösteriminde kullanılan bar-chartlar, çok fazla kategori olması durumunda histogramlardaki gibi istenilen sonuçları vermezler. Kategorilere yönelik sorgularda, kategorileri sayılarına veya önem derecelerine göre sıralamak, küçük grupları tek bir kategori haline getirmek üzere birleştirerek interaktif gösterim tekniklerinin kullanılması gerekir (Unwin,2000).

İstatistiksel gösterim tekniklerinin bütün bu dezavantajlarından kurtulmak için büyük veri setlerinde uygulanmak üzere farklı disiplinlerce farklı gösterim teknikleri geliştirilmektedir. Bilgisayar bilimleri, yeni algoritmalar ve daha hızlı görselleştirme sağlayacak programlar üzerine yoğunlaşmıştır. Büyük veri setlerini görselleştirmeye yarayan programlar henüz yeteri kadar gelişmemiştir. Bu programlar sadece verileri ekran üzerinde grafik bir alanda görselleştirmeye yöneliktir. Özellikle iş dünyasının taleplerine cevap vermek için kullanıcının istediği gibi müdahale edebileceği, interaktif gösterim teknikleri geliştirilmekte ve bu programlara dahil edilmektedir.

İstatistikte ise gösterim tekniklerinin geliştirilmesi daha öncesine dayanır ve bir noktadan sonra diğer disiplinlerle kesişir. Bu süreçte atılan ilk adımlar Tukey'in Exploratory Data Analysis (1977) isimli veri analizi kitabı ve Asimov'un çok değişkenli veri setlerinin izdüşümünü alarak 2 boyutta gösterilmesini sağlayan Grand Tour tekniğini ilk kez olarak ortaya attığı (1985) makalesi sayılabilir.

80'li yılların ortalarından sonra bilgisayar teknolojisinin hızlı gelişmesi ve özellikle uydulardan gelen verilerin çok büyük boyutta olması, görselleştirme üzerine yapılan çalışmalara farklı bir boyut kazandırmıştır. Paralel Koordinatlar, Worlds Within Worlds, Dimension Stacking, Hierarchical Axes,

Hyperbox gibi teknikler bu dönemde geliştirilmiştir (Wong and Bergeron, 1995).

Yakın dönemde gerçekleştirilen veri gösterimi konulu çalışmalar ise, daha önceden varolan teknikleri birleştirmeye yöneliktir. Panel ve Saçılım grafiği matrisini (Scatterplot Matrix) birleştiren HyperSlice, Worlds Within Worlds tekniğini kurala dayalı sorgu (rule-based query) arayüzü kullanarak geliştiren AutoVisual, Dimension Stacking, saçılım grafiği matrisi, Glyphs ve Paralel Koordinatlar'ı çok boyutlu boyama (brushing) uygulayarak tek bir sistemde toplayan XmdvTool en son geliştirilen tekniklerdir. Bu alanda gelinen son nokta; kurala dayalı sorgu kavramının ortaya atılması ve grafiklerin sesle birleştirilmesidir (Wong and Bergeron, 1995).

Veri setlerinin gösterimindeki sınırlamalar, bugünkü teknolojinin yapımına elverdiği monitörlerin çözünürlüğüyle bağlantılıdır. 19 inçlik 1024x1280 çözünürlükteki bir monitörde yaklaşık olarak 1.3 milyon piksel vardır. Ayrıca insan algısı, gösterim tekniklerini uygulayan programlar yaratılırken gözönünde bulundurulan en önemli etkendir. Farklı renk skalaları, parlaklık, koyuluk, derinlik gibi değişkenler üzerinde çalışılarak algılanma açısından en iyi renk modeli yaratılmaya çalışılmaktadır.

6.2. Gösterim Tekniklerinin Sınıflandırılması

Gösterim tekniklerinin sınıflandırılmasında, farklı disiplinler farklı yaklaşımlarda bulunur. Gösterimin amacına göre, verinin tipine ve/veya boyutuna göre, gösterim tekniğinin boyutuna göre, interaktif olup olmamasına göre vb. birçok farklı şekilde sınıflandırma yapılabilir. Veri madenciliğinde gösterim tekniklerinin sınıflandırılmasında 3 ayrı kriter gözününde bulundurulur. Bu kriterler birbirleriyle ortak kullanılabilirler.

6.2.1. Veri tipi / boyutuna göre gösterim teknikleri

- Tek boyutlu veriler: Veri madenciliğinde kullanılan tek boyutlu veriler genellikle zaman serileridir.
- İki boyutlu veriler: En genel şekli boyutları enlem ve boylam olan haritalar ve coğrafik veritabanlarındaki verilerdir. Bu tür verilerin gösterimleri x-y grafiğinin özel bir türü ile gerçekleştirilir.
- Çok boyutlu veriler: Çok boyutlu veriler ilişkisel veritabanlarından elde edilir. Üçten fazla boyut söz konusu olduğunda istatistikte kullanılan 2 ve 3 boyutlu grafikler kullanılmazlar.
- Metin ve web sayfası (hypertext): Yazılı belgelerin ve web sayfaları üzerindeki bilgilerin görüntülenmesidir. Bu veri tipleri sayısal olmadıklarından görselleştirilmeleri için bazı dönüşümlerin yapılması gerekir. En basit dönüşüm kelime sayımıdır. Bu tür dönüşümler genellikle temel bileşen analizi veya çok boyutlu ölçekleme ile görselleştirilir.
- Hiyerarşik yapılar (hierarchies) ve grafikler: Bir şirket içinde e-mail şeması, organizasyon şeması, insanların alışveriş alışkanlıkları, harddisklerin dosya yapıları, bir web sayfasının linkleri hiyerarşik veri tipleridir. Bu tip veriler için özel gösterim teknikleri vardır.
- Algoritma ve programlar: Bir programın anlaşılması veya program geliştirmede yardımcı olmaya yönelik gösterim tekniklerdir. Bir programda bilgi akışını görselleştirmek, programın kod yapısını görselleştirmek, hataları görselleştirmek gibi.

6.2.2. Gösterim tekniklerine göre

Sadece isimleri verilen gösterim teknikleri bölüm 6.3.' de daha ayrıntılı olarak incelenecektir.

- Geometrik gösterim teknikleri: Scatterplots, Landscapes, Projection pursuit, Prosection views, Hyperslice, Parallel coordinates
- Sembolik gösterim teknikleri: Chernoff faces, Stick figures, Shape coding, Color icons, Tilebars

- Hiyerarşik gösterim teknikleri: Dimenson stacking, Worlds within worlds, Treemap, Conetrees, Infocube
- Piksel gösterim teknikleri: Recursive pattern technique, Circle segments, Spiral axes
- Grafik esaslı gösterim teknikleri: Basic graphs, Specific graphS, Systems
- Karma gösterim teknikleri: Farklı gösterim tekniklerinin birleştirilerek kullanıldığı tekniklerdir.

6.2.3. Etkileşim ve bozulum teknikleri

Etkileşim teknikleri; kullanıcının grafiklere istediği yönde müdahale ederek değiştirmesini, birbirinden bağımsız farklı grafikleri birbirine bağlamasını ya da eklemesini sağlayan tekniklerdir.

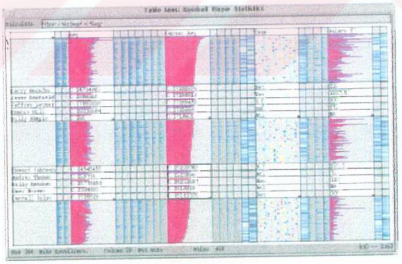
Bozulum teknikleri ise veri setini kullanıcının ayrıntılar üzerine odaklanmasını sağlayacak biçimde görselleştirir. Veri setinin belli bir bölümü yüksek çözünürlükle ayrıntılı bir şekilde görselleştirilirken, diğer bölümleri hangi grafik kullanılıyorsa, onun genel gösterim şekli kullanılarak görselleştirilir.

- Dinamik izdüşüm tekniği (dynamic projections): Veri setinin izdüşümlerini otomatik olarak değiştirerek veri setindeki ilginç yapıları saçılım grafikleri şeklinde rasgele, elle, önceden hesaplanmış ya da veriye göre belirlenmiş bir sırayla gösteren Grand Tour tekniği, dinamik izdüşümlere örnektir.
- İnteraktif filtreleme (interactive filtering): İnteraktif olarak veri setini bölümlere ayırmak ve farklı yapıları incelemek için kullanılan tekniklerdir. Farklı yapılar ya doğrudan bilinerek browsing ile ya da belirlenen niteliklere göre sorgulamayla araştırılabilir. Çok büyük veritabanlarında her iki yöntem de optimal bir şekilde işlemediğinden interaktif Magic Lenses geliştirilmiştir. Magic Lenses 'da veri seti üzerinde gezdirilen sanal bir büyüteç filtre görevi görür ve altındaki veriyi interaktif olarak büyüterek ayrıntıları gösterir.

Şekil 6.1.
Magic Lenses' ın şematik gösterimi
(Keim, 1997)

Filter-Flow Model, InfoCrystal, DEVise, Dynamic Quaries diğer filtering tekniklerdir.

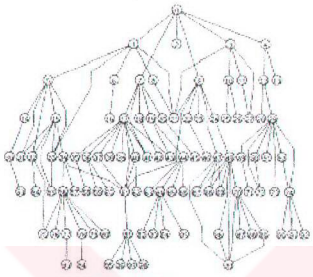
- İnteraktif odaklama (interactive zooming): Veri seti üzerinde interaktif odaklama yapmaya yarayan tekniklerdir. odaklamanın seviyesi arttıkça otomatik olarak verinin görselleştirilmesi de değişir. Düşük seviyelerde piksel olarak görüntülenen veriler, seviye arttıkça ikon ve nesne olarak görselleştirilir. TableLens bu tekniğe örnektir. TableLens veritabanındaki her gözlemi çubuklar şeklinde görselleştirilir. Çubukların yükseklikleri 1 piksel boyutunda olup uzunlukları değişkenine göre belirlenir. Bu sayede bir bakışta veritabanındaki örüntü, ilişki ve aykırı değerler gözlemlenebilir.



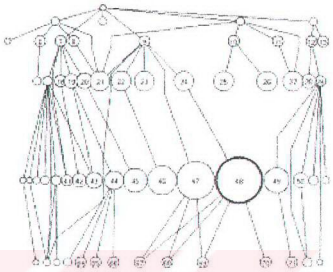
Şekil 6.2.
TableLens örneği
(Keim, 2001)

- İnteraktif bozulum (interactive distortion) teknikleri: En fazla kullanılan interaktif bozulum teknikleri, hiyerarşik veriler ve grafikler üzerinde uygulanan hiperbolik ve küresel bozulum tenikleridir. Bifocal Displays, Perspective Wall, Graphical Fisheye Views, Hyperbolic Visualization ve Hyperbox interaktif bozulum tekniklerine örnektir. Şekil 6.4.' de, Şekil

6.3.'deki orijinal gafik üzerinde Fish Eye bozulum tekniğinin uygulanması gösterilmiştir (Keim, 1999).



Şekil 6.3.
Orijinal grafik



Şekil 6.4.
Fish eye tekniğinin uygulanması

- İnteraktif bağlantı (interactive linking) ve boyama(brushing) teknikleri: İnteraktif bağlantı ve boyama tekniklerinin amacı farklı gösterim tekniklerini birbirine bağlayarak tek bir tekniğin zayıf olan yönünü kapatmaktır. Birçok gösterim tekniği üzerinde uygulanırlar.

6.3. Gösterim Teknikleri

6.3.1. Geometrik gösterim teknikleri

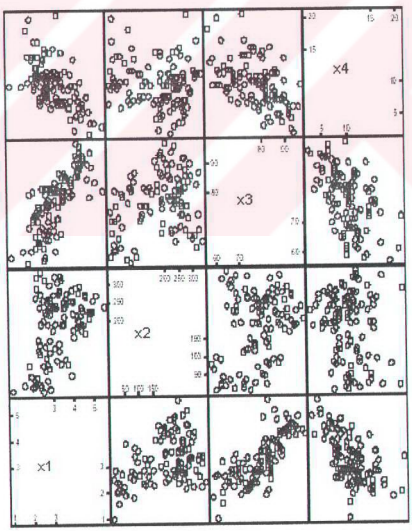
Geometrik gösterim teknikleri, çok boyutlu veri setlerinin 2 boyutta gösterilmesini sağlamak için uygulanan geometrik dönüşümleri ve veri setinin farklı boyutlar üzerine düşen izdüşümlerinin optimal olarak seçimine dayanan teknikleri kapsamaktadır.

Geometrik gösterim tekniklerine örnek olarak saçılım grafiği matrisi, Landscapes, İzdüşüm Takibi Teknikleri, Prosection Views, Hyperslice, Hyperbox, Paralel Koordinatlar, Dairsel Paralel Koordinatlar, Hiyerarşik Paralel Koordinatlar, Grand Tour, Gridviz, Survey Plots, Radviz, Andrew Eğrileri, Permutasyon Matrisi ve Natural Scene Paradigm sayılabilir.

6.3.1.1. Saçılım grafiği matrisi (Scatterplot matrix)

Saçılım grafiklerinin kullanımı çok öncelere dayansa da, saçılım grafiği matrisi ilk olarak Chambers, Cleveland, Kleimer ve Tukey'in 1983'de yayımladıkları "Graphical Methods for Data Analysis" kitabında ortaya atılmıştır. İstatistikte çok değişkenli/boyutlu verilerin görselleştirilmesinde kullanılan en yaygın grafik türüdür (Wong and Bergeron, 1995).

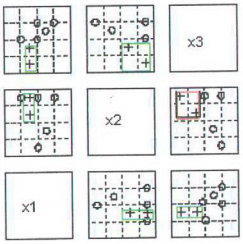
Saçılım grafiği matrisi, tüm değişkenlerin ikişerli ikişerli ve ardışık olarak saçılım grafiklerinin simetrik bir matris oluşturacak şekilde sıralanmasıyla oluşturulur. Köşegenler boş bırakılarak değişken isimleri buraya yazılır. Saçılım grafiklerinin bu şekilde tasarlanmasının nedeni, değişkenler arasındaki bağımlılığı görsel olarak vermektir.



Şekil 6.5.
4 değişkenli bir saçılım grafiği matrisi
(McLeod and Provost, 2001)

Statik bir saçılım grafiği matrisi, sadece birkaç yüz veriye kadar rahat görselleştirme yapabilir. Bu dezavantajını gidermek için üzerinde boyama (brushing) tekniği uygulanır.

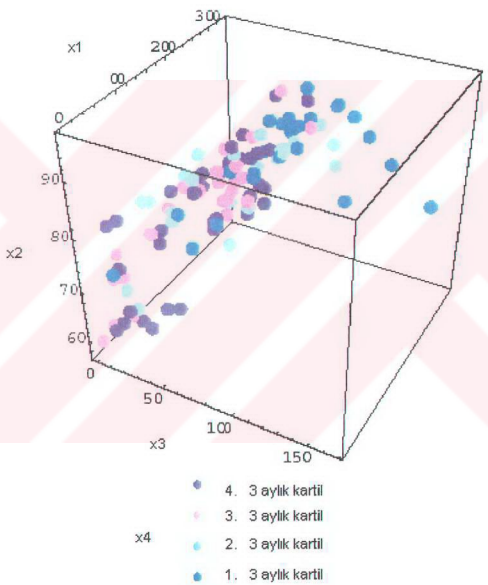
Boyama tekniği, ilk olarak 1987 yılında Becker ve Cleveland'ın Technometrics'de yayımladıkları bir makale ile ortaya atılan, bugüne kadar geçerliliğini koruyan ve birçok farklı gösterim tekniği üzerinde uygulanan interaktif bir tekniktir. Labeling ve enhanced linking olmak üzere 2 türlü boyama tekniği vardır. Labeling; saçılım grafiği matrisi içinde, herhangi bir saçılım grafiği üzerinde farklı yapıda olan verilerin mouse gibi bir işaretliyiciyle boyayarak belirlenmesidir. Boyanan veriler eşanlı olarak diğer saçılım grafiklerinde de boyanacaktır, böylelikle verilerin nasıl değiştiği gözlenebilir. Enhanced linking ise farklı yapıda olan verilerin bir işaretleyiciyle dikdörtgen içine alınmasıdır. Diğer saçılım grafiklerinde bu dikdörtgenin yapısının nasıl değiştiği, verilerin konumu hakkında görsel olarak bilgi verir. Şekil 6.6.' de 3 boyutlu bir saçılım grafiği matrisinin enhanced linking ile boyanması gösterilmektedir. Kırmızı dikdörtgen içine alınan farklı yapıda verilerin, diğer saçılım grafiklerinde değişimi görünmektedir. (http://catt.bus.okstate.edu/jones98/brushing.htm)



Şekil 6.6.
Enhanced Linking

Saçılım grafiği matrisinin kullandığı değişkenlerin ikişerli ikişerli ve ardışık olarak simetrik bir matris oluşturacak şekilde gösterimi, Hyperbox ve

Hyperslice gösterim tekniklerinde de kullanılır. Ayrıca saçılım grafiği matrisi 3 boyutlu olarak da görselleştirilebilir. Özellikle zamanın 4. değişken olarak kabul edilmesiyle oluşturulan 4 boyutlu grafikler, saçılım grafikleri üzerinde de uygulanabilir. Şekil 6.5.' de kullanılan 4 değişkenli veri setinin 3 boyutlu saçılım grafiği ile gösterimi aşağıda verilmiştir. 4. değişken 3'er aylık kartiller olarak farklı renklerle gösterilmiştir.



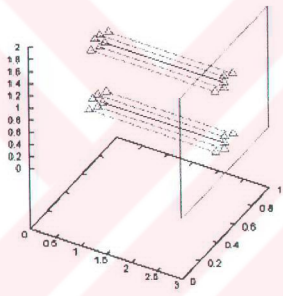
Şekil 6.7.
3 boyutlu saçılım grafiği
(McLeod and Provost, 2001)

6.3.1.2. İzdüşüm Takibi (Projection Pursuit)

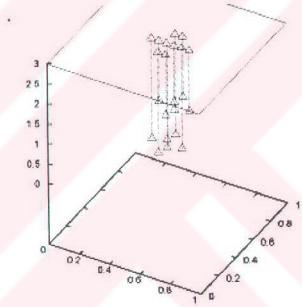
Temel bileşenlerin genelleştirilmiş şekli olan İzdüşüm Takibi, çok boyutlu veri setlerinin 2 boyutlu bir düzlem ya da 3 boyutlu bir hacim üzerindeki izdüşümlerinin gösterimidir. İstatistikte sıkça kullanılan temel bileşenler analizi, faktör analizi ve çok boyutlu ölçekleme birer izdüşüm takibi örneğidir. Kruskal tarafından ortaya atılan (1969), Friedman ve Tukey tarafından

geliştirilen (1974) bu yöntem Huber'in (1985) çalışmalarıyla son halini almıştır.

Veri seti farklı açılardan ele alındığında birden fazla düzlem üzerinde izdüşümü olur. Bazı izdüşümler veri setinin yapısı hakkında bilgi sağlarken, bazıları veri setini açıklamada yetersiz kalır. İzdüşüm takibi, veri setini açıklayan en optimal izdüşümü belirleyen bir optimizasyon problemi olarak görülebilir. Şekil 6.8. ve Şekil 6.9.' da aynı veri setinin farklı izdüşümleri gösterilmektedir (Brunsdon, Fotheringham and Charlton, 2000).



Şekil 6.8.
Bir veri setinin (R^3) sağdaki düzlem üzerine (R^2) izdüşümü. 2 ayrı küme gözleniyor.



Şekil 6.9.
Aynı veri setinin yukarıdaki düzlem üzerindeki izdüşümü. Tek küme gözleniyor.

<u>İzdüşüm Takibi algoritması</u>

j değişkeni, i de gözlemi göstermek üzere $X = \{x_{ij}\}$ bir veri matrisi olsun.

a ve b, lineer dönüşümü gösteren m boyutlu satır vektörleri, z_1 ve z_2 de izdüşüm düzlemindeki noktaları gösteren n boyutlu satır vektörleri olmak üzere R^m den R^2 ye izdüşüm;

$$(z_1, z_2) = (Xa', Xb') \tag{6.1}$$

olarak yazılabilir. Optimal izdüşümün seçimi, artık a ve b nin hesaplanması işlemidir. İzdüşüm vektörleri z_1 ve z_2 birbiriyle ilişkili değildir. Bundan dolayı çok boyutlu veri setinden 2 boyutlu grafiğe maksimum bilgi aktarılır. z_1 ve z_2 ilişkili olsalardı, herbiri ayrı ayrı bir örüntüyü değil, ortak gizli bir örüntüyü paylaşmış olurlardı (Brunsdon and Fotheringham and Charlton, 2000).

Optimal izdüşümün bulunmasında ilk aşama, veri seti üzerinde neyin araştırılacağına karar verilmesidir. Amaca göre bir indeks fonksiyonu (I) seçilir. Eğer kümeleme amaçlanıyorsa, indeks fonksiyonu için genellikle ortalama en yakın komşu uzaklığı (mean nearest neighbour distance, MNND) kullanılır. Bu değer ne kadar küçükse kümeleme o kadar iyi olur.

$I(z_1, z_2)$ olarak gösterilen izdüşümün indeks fonksiyonu MNND' dir. indeks fonksiyonu;

$$I(z_1, z_2) = I(Xa', Xb') \tag{6.2}$$

olarak yazılabildiğinden, izdüşümün seçimi indeks fonksiyonunu minimum kılan a ve b' yi bulmaya yönelik bir optimizasyon problemi olarak düşünülebilir. MNND skaler bir değer olduğundan, a ve b' yi bir sabitle çarpmak, aynı zamanda MNND yi de o sabitle çarpmak anlamına gelecektir. Eğer bu sabit çok küçük seçilirse, MNND istenildiği kadar minimum olacaktır. Bu problemi ortadan kaldırmak için z_1 ve z_2 ortalaması sıfır, varyansı 1 olacak şekilde standartlaştırılır.

İzdüşüm takibi algoritması, kısıtları belli bir optimizasyon problemidir. Fakat dönüşümden dolayı kısıtların a ve b şeklinde değil de z_1 ve z_2 şeklinde olmasından dolayı, çözümü kolaylaştırmak için X veri matrisinin her bir değişkeni, varyansları 1 olan temel bileşenlerine dönüştürülür. Bundan sonraki işlemlere X yerine Q dönüşüm matrisi ele alınarak devam edilir. Q,

X' in lineer dönüşümü olduğundan, X matrisinin izdüşümü, aynı zamamda Q temel bileşenler matrisinin de izdüşümüdür. Buna göre;

$$I(z_1, z_2) = I(Qc', Qd') \tag{6.3}$$

olarak yazılabilir. Bu eşitlikte c ve d dönüşümden dolayı a ve b'nin yerini alan vektörlerdir.

Ortalamaları 0, varyansları 1 olan z_1 ve z_2' nin ilişkili olmaması için $c'c = 1$, $d'd = 1$ ve $c'd = 0$ olmalıdır. Projection pursuit algoritması;

Min $I \quad (Qc', Qd')$

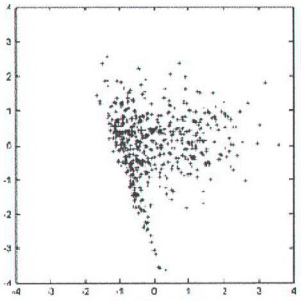
$c'c = 1$

$d'd = 1$ ve

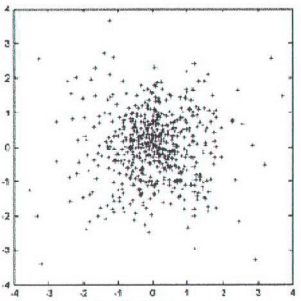
$c'd = 0$ olacak şekilde

açıklanan bir optimizasyon problemine dönüşmüştür. Bu optimizasyon probleminin zorluğu, indeks fonksiyonuna bağlıdır. İndeks fonksiyonunun MNND olması, işlemlerin yoğun olmasına neden olur, çünkü izdüşüm düzlemindeki her bir nokta için en yakın komşunun hesaplanması gerekir.

Eğer amaç kümeleme değil de aykırı gözlemlerin saptanması olsaydı, indeks fonksiyonu MNND' nin maksimizasyonunu gerektirecekti. En yakın komşuya en uzaktaki nokta aykırı gözlem olacağından MNND değerinin en büyük olması gerekecektir. Algoritma da maksimizasyon problemine dönüşecektir. Aşağıda aynı veri setinin MNND'nin minimizasyonu ve maksimizasyonuna göre oluşan izdüşümleri gösterilmektedir (Brunsdon, Fotheringham and Charlton, 2000).



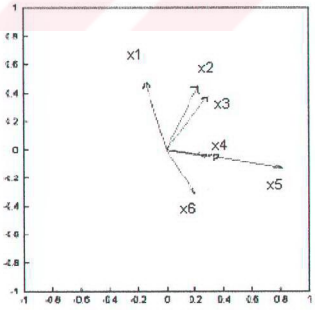
Şekil 6.10.
MNND'nin minimizasyonu



Şekil 6.11.
MNND'nin maksimizasyonu

Interpretation plot

c ve d nin optimizasyonundan sonra, geriye doğru işlem yapılarak a ve b hesaplanabilir. İzdüşüm lineer olduğundan, değişkenlerdeki herhangi bir değişme, (a, b)' yi de değiştirecektir. Interpretation plot, değişkenlerin izdüşüm düzlemindeki bir noktanın tüm değişkenler 1 standart sapma kadar değiştirildiğinde hangi yöne ne kadar gideceğini gösterir. Buna örnek olarak Şekil 6.12.'deki grafik verilmiştir.



Şekil 6.12.
Interpretation plot
(Brunsdon, Fotheringham and Charlton, 2000)

6.3.1.3. Hyperslice

Hyperslice, 1993 yılında J. Wijk ve R. Liere tarafından ortaya atılan çok değişkenli veri setlerini, ikili saçılım grafiklerini matris şeklinde gösteren yöntemlere alternatif olarak, kullanıcı tarafından müdahaleyi kolaylaştıran ve çabuklaştıran interaktif bir geometrik gösterim tekniğidir (Wijk and Liere,1993).

Üçten fazla değişkenli veri setlerinde tüm değişkenlerin aynı anda gösterimi fiziksel olarak imkansız olduğundan görselleştirme değişken seçimi yapılarak gerçekleştirilir. Fakat bu tip görselleştirmede kullanıcının müdahalesi teknik olarak görselleştirme sürecini çok fazla uzatır ve verilerin ekrana gelmesi için ek algoritmalar gerekir. Hyperslice tüm değişkenleri aynı anda gösterebilen ve kullanıcı tarafından değiştirilen parametrelere göre, grafik alanı hızlı bir şekilde yenileyebilen bir teknik olduğundan oldukça avantajlıdır.

c odak nokta olarak adlandırılan ve veri seti içinde özellikle ilgilenilen çok boyutlu nokta, $(c_1, c_2,...,c_N)$,

w_i; odak noktasının dahil olduğu, görselleştirilmede kullanılacak veriler için seçilen genişlik (i = 1,2,..N) olmak üzere,

$R_i = (c_i - (w_i/2), c_i + (w_i/2))$ alanı içindeki veriler için bir f(x) fonksiyonunun aldığı değerleri, Hyperslice, saçılım grafiği matrislerinde olduğu gibi tüm değişken çiftlerini ardışık bir sırayla simetrik bir matris oluşturacak şekilde görselleştirir. Bu matrisin her gözüne panel adı verilir.

Matrisin köşegenlerinde ve köşegen dışında bulunan paneller içinde farklı grafikler bulunur. Köşegen dışındaki panellerde;

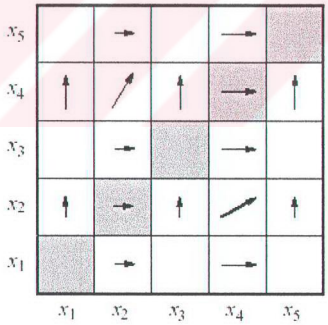
$x_k \in R_k$, $x_l \in R_l$ ve $x_i = c_i$ olmak üzere; $(i \neq k)$, $(i \neq l)$

f(x) fonksiyonunun 3 boyutlu görsel bir ifadesi olan $S_{k,l}$ bulunur. Bu ifadeye Slice denir.

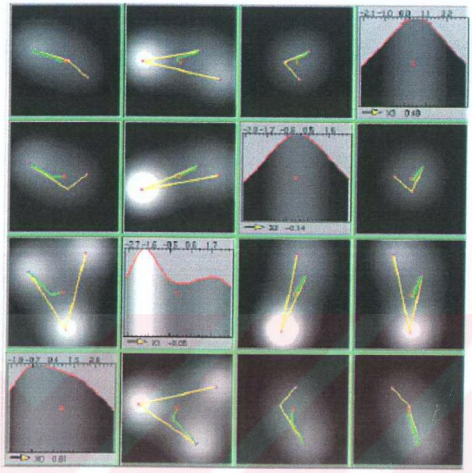
Hyperslice'ın köşegen panellerinde ise

$x_k \in R_k$ ve $x_i = c_i$ olmak üzere f(x) fonksiyonunun tek boyutlu bir grafiği olan G_k yer alır.

Hyperslice' ın saçılım grafiği matrisinden farkı, verilerin kullanıcı tarafından belirlenmiş bir odak nokta etrafında interaktif olarak yönlendirilebilmeleridir. Kullanıcı herhangi bir paneldeki odak noktayı mouse yardımıyla hareket ettirebilir. Bir panelin odak noktasını değiştirmek genellikle veriler üzerinde farklı bir durum gözlendiğinde yapılır. Odak noktasının değişmesi, diğer panellerdeki grafikleri de eşanlı olarak değiştireceğinden, Hyperslice kullanıcıya f(x) fonksiyonu değerleri için ayrıntılı bilgi vermiş olur. Şekil 6.13.'de (4,2) panelindeki odak noktanın ok yönünde kaydırılmasıyla diğer panellerdeki değişimin yönü gösterilmiştir.



şekil 6.13.
(Wijk and Liere,1993)

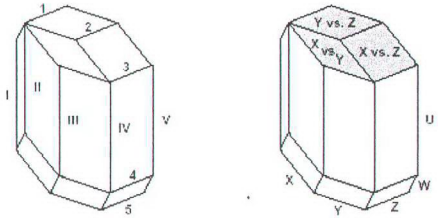


şekil 6.14.
4 boyutlu bir Hyperslice örneği
(Wijk and Liere,1993)

6.3.1.4. Hyperbox

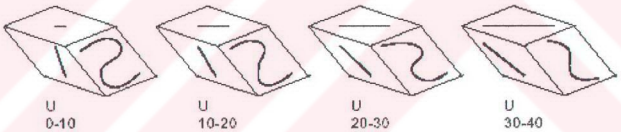
B. Alpern ve L. Carter tarafından 1991 yılında ortaya atılan Hyperbox, saçılım grafiği matrisi ve Hyperslice gibi sıralı olarak ikili değişkenlerin grafiklerini matris şeklinde gösterir. Fakat yapısı gereği Hyperbox diğer iki teknikten ayrılır.

n boyutlu bir Hyperbox'da, n^2 tane doğru ve n(n-1)/2 tane yüz vardır. Her doğru için n-1 tane o doğruyla aynı uzunlukta ve doğrultuda olan doğru vardır. Hyperbox' ı oluşturan doğruların uzunlukları ve eğimleri, veri setindeki değişkenleri ifade eder.



Şekil 6.15.
5 boyutlu bir Hyperbox örneği
(Keim and Kriegel, 1996)

Hyperbox, belirlenen bir değişkene göre kesilerek verilerin değişimi gözlenir. Zaman serileri verileri üzerinde uygulanması görsel açıdan kolay ve çabuk algılanabilir sonuçlar verir. Şekil 6.16.'de bir zaman serisinin zaman aralıklarına göre kesilmiş Hyperbox üzerinde gösterimi verilmiştir. Yüzeyler verilerin zaman aralıklarında değişimini simgelemektedir.



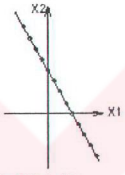
Şekil 6.16.
Kesilmiş Hyperbox
(Keim and Kriegel, 1996)

6.3.1.5. Paralel koordinatlar (Parallel Coordinates)

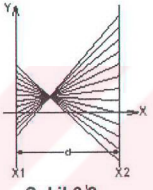
Inselberg tarafından ortaya atılan ve Wegman tarafından geliştirilen Paralel Koordinatlar, çok değişkenli veri setlerinin iki boyutlu grafiklerde görüntülenmesini sağlar. Sadece verilerin çok boyutluluğuyla değil, nesnelerin geometrisiyle de ilgili olduğundan diğer gösterim tekniklerinden ayrılırlar.

n değişkenli bir veri setinin sadece iki değişkeni, iki eksenli kartezyen koordinatlar üzerinde aynı anda gösterilebilir. Bütün değişkenlerin aynı anda

iki boyutlu grafiklerle gösterilebilmesi için, koordinat sisteminin değiştirilmesi gerekir. Değişkenler, aralarında belirli bir uzaklık olan ve birbirine paralel eksenler olacak şekilde tasarlandığında, çok boyutlu veri setlerinin iki boyutlu grafikler üzerinde görselleştirilmesi sağlanmış olunur. Kartezyen koordinatlardaki değişkenler paralel koordinatların eksenleri olduklarından, kartezyen koordinatlar üzerindeki her bir nokta, paralel koordinatlarda doğrular ile ifade edilir.



Şekil 6.17.
Kartezyen Koordinatlar



Şekil 6.18.
Paralel Koordinatlar

ℓ : kartezyen koordinatları, $\overline{\ell}$ paralel koordinatları simgelesin.

$\ell : y = mx + b$ kartezyen koordinatlarda bir doğruya denk gelecektir. Bu doğru üzerinde (a,ma+b) ve (c,mc+b) olmak üzere iki nokta ele alınsın. Bu noktalar eksenleri t ve u olan paralel koordinatlar üzerine taşındığı zaman kartezyen koorinatta (a,ma+b) noktası, paralel koordinatta t ekseni üzerinde (a,0) noktasını u ekseni üzerindeki (ma+b,1) noktasına bağlayan doğruya denk düşecektir. Aynı şekilde kartezyen koordinatta (c,mc+b) noktası, paralel koordinatta t ekseni üzerinde (c,0) noktasını u ekseni üzerindeki (mc+b,1) noktasına bağlayan doğru olacaktır.

Paralel koordinattaki iki doğru ($b(1-m)^{-1}$, $(1-m)^{-1}$) noktasında kesişecektir. Bu kesişim noktası kartezyen koordinattaki doğrunun b ve m parametrelerine bağlıdır. ℓ, $\overline{\ell}$' in dualidir. Yapılan dualite işleminden sonra, kartezyen koordinattaki noktalar paralel koordinatta doğrulara, paralel koordinattaki doğrular da kartezyen koordinattaki noktalara denk düşer.

$\ell : y = mx + b$ doğrusu yapılan dualite işleminden sonra,

$\overline{\ell} : \left(b(1-m)^{-1}, (1-m)^{-1}\right)$ noktasına denk gelecektir.

$0<(1-m)^{-1}<1$ arasında m negatiftir ve kesişim paralel koordinatların eksenlerinin arasında gerçekleşir. m=-1 için kesişim eksenlerin tam orta noktasındadır. $(1-m)^{-1}< 0$ veya $(1-m)^{-1} > 1$ durumunda, m pozitiftir ve kesişim paralel eksenlerin dışında gerçekleşir. m=1 olması özel bir durumdur. Bu durumda formül geçersiz olur. Yüksek ilişkili verilerde ise paralel koordinatlar arasında kesişmeyen doğrular oluşur.

ℓ kartezyen koordinatları $y = mx + b$ doğrusu için $(m, -1, b)$ şeklinde gösterilebilir. Bu doğrunun $\overline{\ell}$ paralel koordinatlarda denk düşeceği noktalar ise $\left(b(1-m)^{-1}, (1-m)^{-1}, 1\right) = (b, 1, 1-m)$'dir.

t ; paralel koordinat ekseni, x kartezyen koordinat ekseni olmak üzere; Paralel Koordinatlar;

$$t = xA \tag{6.4}$$

şeklinde bir dönüşümü görselleştirir. Bu dönüşüm nokta cinsinden yazıldığında;

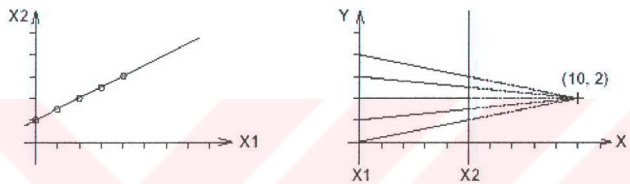
$$(b, 1, 1-m) = (m, -1, b)A \quad \text{elde edilir.} \tag{6.5}$$

Burada A kartezyen koordinattaki doğruya bağlı olarak değişen bir matristir:

$$A = \begin{bmatrix} 0 & 0 & -1 \\ 0 & -1 & -1 \\ 1 & 0 & 0 \end{bmatrix}$$

Yapılan bu dönüşüm, kartezyen koordinatların doğrularını paralel koordinatlarda noktalara dönüştürür (Wegman and Carr, 2001).

Paralel koordinatlar, özellikle 3 boyutlu zaman serilerini kartezyen koordinatlardan çok daha iyi görselleştirirler. Kesişim noktalarının pozisyonu, her ikili değişken arasındaki ilişkiye dair kabaca bir bilgi verir.

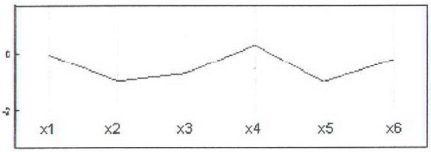


Şekil 6.19.
Paralel Koordinatta kesişim noktaları
(Keim and Kriegel, 1996)

Bu tekniğin dezavantajı, paralel doğrular arasında limitli bir uzaklık olması ve gösterebildiği veri sayısının belirli olmasıdır.

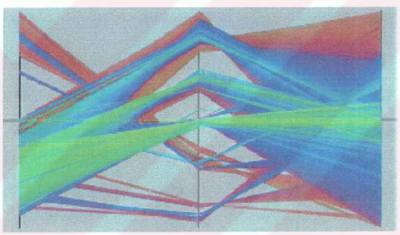
Kartezyen uzaydan paralel koordinatlara geçişte biraz bilgi kaybı olur ama bu problem doğruların yerlerinin sıralanmasıyla giderilebilir. Değişkenlerin, dolayısıyla paralel doğruların farklı sıralanışı farklı paralel koordinat grafikleri yaratır. Paralel Koordinatlar'ın interaktif olması, doğruların yeniden sıralanmasının hızlı ve esnek bir şekilde gerçekleşmesini sağlar. Aynı zamanda veri seti büyük olunca alt kümelerin farkedildiği durumlarda, bu kümeleri mouse yardımıyla seçebilmek, farklı grupların yerlerini değiştirmek, Paralel Koordinatların sağladığı kolaylıklardır. Paralel Koordinatlar üzerinde boyama tekniği sıkça kullanılır (Unwin, Volinsky and Winkler).

Şekil 6.20'de kartezyen koordinatlarda tek bir noktanın paralel koordinatlarda görüntüsü verilmiştir.



Şekil 6.20.
Tek veriden oluşan bir Paralel Koordinat
(Brunsdon, Fotheringham and Charlton)

Şekil 6.21. ise üzerinde boyama tekniği uygulanarak farklı yapıdaki verilerin açığa çıkması sağlanan Paralel Koordinatlar verilmiştir.



Şekil 6.21.
Paralel Koordinatlar
(Keim, 2001)

6.3.2. Sembolik gösterim teknikleri

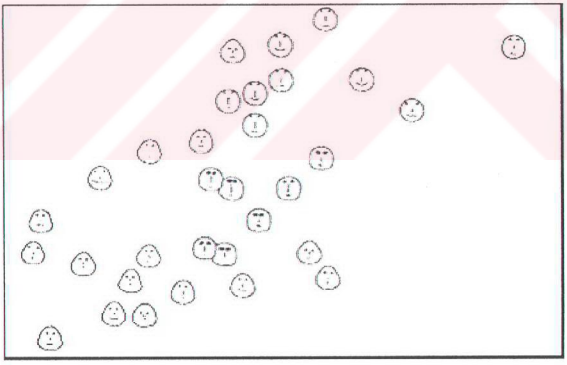
Sembolik gösterim teknikleri, çok boyutlu verilerde değişkenleri sembollerdeki değişimle görselleştiren tekniklerdir. Kullanım amaçlarına göre farklı tasarlanan sembolik gösterim teknikleri, ya veri setinin keşfi ya da sadece verileri görselleştirmek için kullanılırlar. Chernoff Yüzleri (Chernoff faces) gibi verilerin görselleştirilmesini sağlayan teknikler, veri setindeki bir etkiyi (genellikle değişkenler alınır) semboller üzerine taşıyarak görselleştirmeyi gerçekleştirir. Verileri keşfetmeye yönelik teknikler ise grafik alan içerisine mümkün olduğunca çok değişken ve veriyi, yansız ve birbirinin üzerine gelmemesi için bağımsız olarak yerleştirecek şekilde tasarlanmışlardır. (Wegman and Carr, 2001).

Sembolik gösterim tekniklerin dezavantajları, çok fazla veriyi görselleştirememeleridir. İnsan algısı bir bakışta birkaç değişkenin görselleştirildiği grafikleri algılayabilir. Ayrıca veri seti büyüdükçe, sembollerin üstüste gelmesi sözkonusudur.

Sembolik gösterim tekniklerine örnek olarak Chernoff Yüzleri, Çubuk Figürü, Renk İkonları, Sound Icons, Star Glyphs, Driftweed, Dimensional Stacking, TileBars verilebilir.

6.3.2.1. Chernoff yüzleri (Chernoff faces)

Chernoff tarafından 1979'da ortaya atılan Chernoff Yüzleri, veri setinin değişkenlerinden istenilen ikisini grafik ekseni olarak alır. Diğer değişkenler ise insan yüzleri üzerinde göz, ağız, kulak, ve/ya burnun değişimi olarak görselleştirir. Az sayıda veri görselleştirebilirler.



Şekil 6.22.
Chernoff Yüzleri
(Ankerst, 2000)

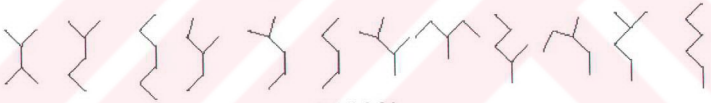
Chernoff yüzleri ile yüz genişliği, ağız uzunluğu, ağız kenarların şekli, gözbebeği pozisyonu, kaş kalınlığı, uzunluğu ve açısı, burun kalınlığı ve genişliği, kulak pozisyonu, gözlerin birbirine ve çeneye olan uzaklığı gibi 30'a yakın değişken görüntülenebilir.

6.3.2.2. Çubuk Figürü (Stick Figures)

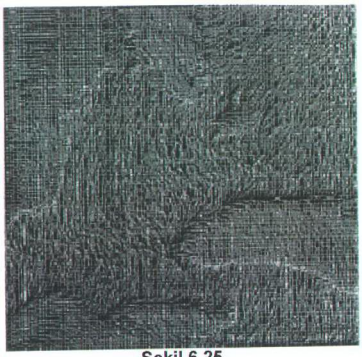
Pickett ve Grinstein tarafından ortaya atılan bu teknik, veri setini birbirlerine bağlandıkları açıları kontrol edilebilen biri beden 4'ü kol olmak üzere 5 doğrudan oluşan sembollerle görselleştirir. Şekil 5.23.'de bir çubuk figürü ve α, β, χ, δ, ε açıları gösterilmektedir.

Şekil 6. 23.
Çubuk figürü açıları
(Brunsdon, Fotheringham and Charlton)

Kolların her biri bir değişkeni simgeler. Daha fazla değişkenin olması durumunda, kolların uzunlukları, kalınlıkları, açıları ya da renkleri ilave edilebilir. 30 değişkene kadar kullanıldıkları durumlar vardır.



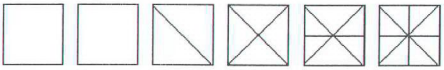
Şekil 6.24
Çubuk figürleri seti
(Brunsdon, Fotheringham and Charlton)



Şekil 6.25.
Bir veri setinin çubuk figürleri ile gösterimi
(Keim,1997)

6.3.2.3. Renk İkonları (Color Icons)

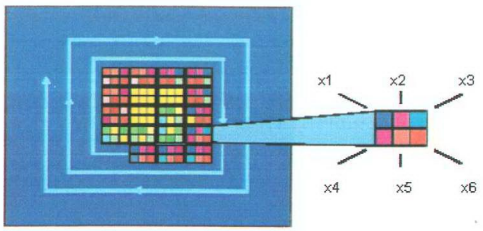
Levkowitz tarafından ortaya atılan Renk İkonları, renk, biçim, boyut, yön, sınır ve bölümlendirme biçimleri olarak değişkenleri gösterir. Şekil 6.26.'da farklı sayıda değişkenleri gösteren Renk İkonları gösterilmektedir.



Şekil 6.26.
Renk İkonlarının değişkenleri
(Wong and Bergeron, 1995)

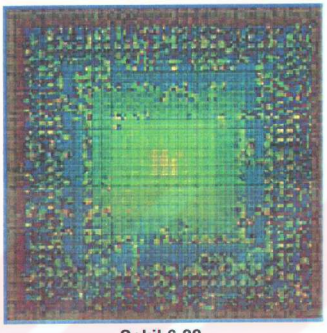
Bir renk ikonunu boyamak için iki yol vardır. Color shading denilen ilk yöntemde, renkli ikonunu oluşturan doğrulara, denk düştükleri değişken değerine göre bir renk atanır. Daha sonra interpolasyonla ikonun rengi belirlenir. İkinci yöntem ise, ikonu oluşturan doğrularla bölünmüş olan alt alanların, denk düştükleri değişken değerlerine göre renklendirilmeleridir.

Şekil 6.27'de 1 gözlemin 6 değişken için aldığı değerlerin farklı renklerle ifade edilmesi gösterilmiştir. Bütün gözlemler renk ikonları ile görselleştirildikten sonra, grafik alan içerisine farklı yöntemlerle yerleştirilirler. Aşağıdaki örnekte renk ikonları grafik alanın orta noktasından başlayan ve dikdörtgen çizecek şekilde kendi çevresinde dönerek ilerleyen bir yol çizerek grafik alanı doldurmaktadırlar.



Şekil 6.27.
Renk ikonları
(Keim, 1997)

Şekil 6.28.'de renk ikonlarıyla görselleştirilmiş bir veri seti gösterilmektedir. Veri setindeki kümeler rahatlıkla ayırdedilebilmektedir.



Şekil 6.28.
Renk ikonlarıyla kümelerin gösterilmesi
(Keim, 1997)

6.3.2.4. TileBars

Bir arama motorundan belirli bir konu arandığında, motor bu anahtar kelimeyi web sayfalarının önceden tanımlanmış olan başlık ve özetlerden tarayarak ulaşır. Information retrieval, başlıklardan ve özetlerden oluşan metin türü veriler üzerine (metadata) çalışan bir daldır. TileBars ise; tüm metinden değil, metin üzerinde uyguladığı sorgulardan yola çıkarak metnin yapısı hakkında bilgi edinmeyi amaçlar.

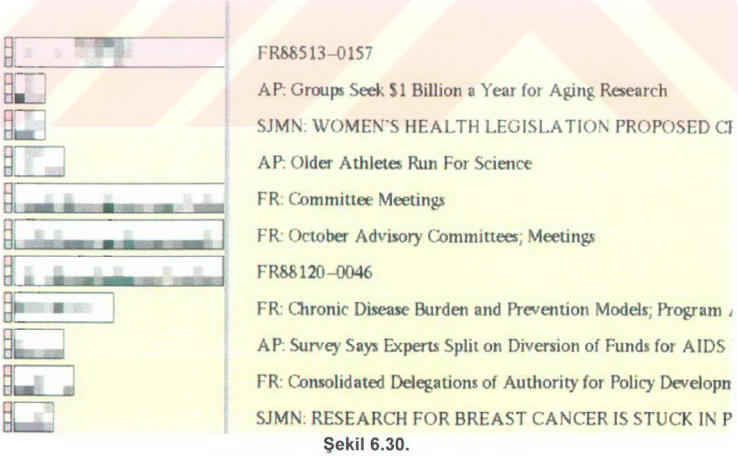
Hearst tarafından geliştirilmiş olan TileBars, eşanlı olarak bağıl metin uzunluğu, sorgulanan terimin frekansı ve metnin içindeki dağılımı hakkında bilgi verir. TileBars, 3 hipotez üzerine kuruludur.

1. Uzun dokümanlar, kısa dokümanlar ve özetler, içlerinde geçen terimin frekansı bakımından birbirlerinden farklıdırlar.

2. Araştırılacak dokümanlar ve sorgulanan terim arasındaki ilişki, kullanıcı tarafından kısa, tutarlı ve doğru olarak tanımlanmıştır.
3. Doküman içindeki pasajlar, dokümanın kendi içindeki konuya ve sorgudaki terime içerik sağlayacak şekilde tasarlanmıştır (Hearst, 1995).

TileBars, bir dokümanı parçalara ayırır. Bu parçalara alt konu (subtopic) denir. Her alt konu, TileBars' da bir satıra denk gelir. Satırlar da kendi içinde parçalara ayrılmıştır. Sorgulanan terimin frekansına göre bu parçalar gri tonlarına göre renklendirilir. Böylelikle doküman hakkında görsel bilgi edinilmiş olunur. Basit bir TileBars örneği aşağıda verilmiştir. Örnekte birinci ve ikinci alt konuların paylaştığı çok fazla ortak terim vardır.

Şekil 6.29.
TileBars



Şekil 6.30.
Bir dokümanın TileBars ile gösterimi
(Hearst, 1995)

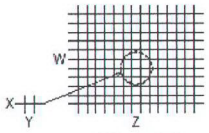
6.3.3. Hiyerarşik gösterim teknikleri

Hiyerarşik gösterim teknikleri, k boyutlu veri setini bölümlere ayırarak 2 ya da 3 boyutlu grafiklerle hiyerarşik bir düzen içinde görselleştirir. Bu teknilere örnek olarak Cone Trees, Debdograms, Dimesional Stacking (General Logic Diagrams), Fractal Foam, Hyperbolic Trees, Infcube, Sunburst, Hiyerarşik Eksenler ve Treemap verilebilir.

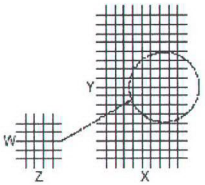
6.3.3.1. Dimensional Stacking

Dimensional Stacking, Hierarchical Axes'in bir türevidir. Her bir gözlem 2 boyutlu xy grafiği olarak kabul edilir. Bu grafiklerin eksenleri hızlı veya yavaş eksen olarak nitelendirilirler ve değişkenler hızlarına göre grafiklere yerleştirilirler. Değişkenlerin hızlarının tanımlanmasında farklı yöntemler kullanılır. Değişkenlerin minimum değer/maksimum değer oranı, ortalama/standart sapma oranı değişkenlerin hızları olarak alınabilir. Değişken sayısının tek olması durumunda bir gölge değişken eklenir (Wong and Bergeron, 1995).

Örneğin x, y, z ve w bağımsız değişkenlerine karşılık gelen büyüklükler sırasıyla 2, 3, 5 ve 6 olsun. x ve y' nin hızlı değişkenler olmaları durumunda Şekil 6.31.'deki grafik elde edilecektir. z ve w' nin hızlı değişkenler olmaları durumunda ise, Şekil 6.32.' deki grafik elde edilecektir.

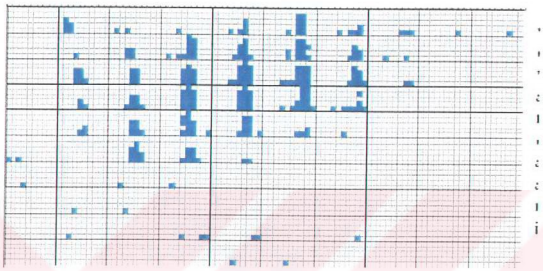


Şekil 6.31.
x ve y'nin hızlı olmaları durumu



Şekil 6.32.
z ve w'nin hızlı olmaları durumu

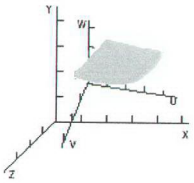
Dimensional Stacking'de bağımlı değişkenler, değerlerine göre farklı bir renk alarak grafikler içinde işaretlenirler. Şekil 6.33.'de bir bağımlı değişkenin renklendirilmiş olarak görselleştirildiği Dimensional Stacking grafiği verilmiştir.



Şekil 6.33.
(Keim, 1997)

6.3.3.2. Worlds Within Worlds

Worlds Within Worlds, değişkenleri içiçe geçmiş 3 boyutlu grafikler üzerinde görselleştiren interaktif bir hiyerarşik gösterim tekniğidir. Sanal bir eldiven yardımıyla içiçe geçmiş grafiklerin yerleri değiştirilir. Kullanıcıya veri setini farklı açılardan gösteren bu tekniğin geliştirilmesine devam edilmektedir. AudioVisual; Worlds Within Worlds üzerine kurala dayanan arayüz eklenerek görselleştirme yapan bir tekniktir. Şekil 6.34' de 6 değişkenli bir Worlds Within Worlds gösterilmektedir.

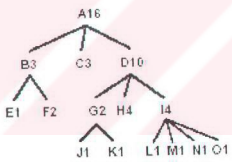


Şekil 6.34.
Worlds Within Worlds
(Wong and Bergeron, 1995)

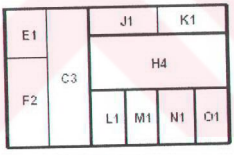
6.3.3.3. Treemap

Ağaç yapıları, hiyerarşik verileri "birleştirerek" ya da "dahil ederek" görselleştirirler. Ağaç diyagramları bu yapılardan en sık kullanılan ve en basit olanıdır. (Barbaria, 2001)

Ağaç diyagramlarında alt dallar organize edilirken grafik alan optimal bir şekilde kullanılmaz, ayrıca alt dallar büyüdükçe ağaç diyagramlarının gösteriminde zorluk yaşanır. Bu dezavantajları yoketmek için Shneiderman tarafından ortaya atılan Treemaps, hiyerarşik verileri 2 boyutlu dörtgenler olarak ve uzay doldurma (space filling) mantığıyla grafik alana yerleştirdiğinden grafik alanı %100 kullanır. Dörtgenlerin boyutları ve renkleri değişkenlere denk gelir. Şekil 6.35.'de harfler alt grupları, rakamlar ise ağırlıkları göstermek üzere bir ağaç diyagramı verilmiştir. Ağaç diyagramının uygulandığı veri setinin Treemap ile görselleştirilmesi Şekil 6.36.'dadır. Treemap'de gruplar ağırlıklarına göre alana yerleştirilmektedir.



Şekil 6.35.
Ağaç diyagramı



Şekil 6.36.
Treemap

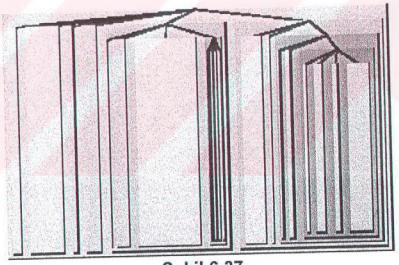
Treemaps, hiyerarşik verileri grafik alana yerleştirirken 2 farklı algoritma kullanır.

1.Top-down algoritması

Top-down algoritması, genel ağaç diyagramlarının mantığını yansıtır. Grafik alanı ekranın bir tarafından başlayarak aksi tarafına doğru dörtgenlere ayırarak ilerler. Bu ayırma işlemi, ağacın alt gruplarının önceden bilinen ağırlıklarına göre yapılır. Top-down algoritmasının aşamaları:

1. Grafik alan en büyük dörtgendir ve ilk grup (root node) olarak kabul edilir. Bölme işlemi bu dörtgenden başlar.
2. İlk grup, dikey olarak kendi ağırlığı, kendinden sonra gelen tüm alt grupların toplam ağırlıyla oranlanarak bölünür. Böylelikle bir üst grup, ve daha sonra bölünmek üzere bir alt grup oluşur.
3. Oluşan alt grup, yatay olarak tüm alt grupların ağırlıklarına göre ayrılır ve birbirleriyle karışmamaları için herbiri ayrı kutular içinde gösterilir.
4. 2. adıma geri dönülür, oluşan her bir alt grubu ilk grup kabul ederek bölme işlemine devam edilir.

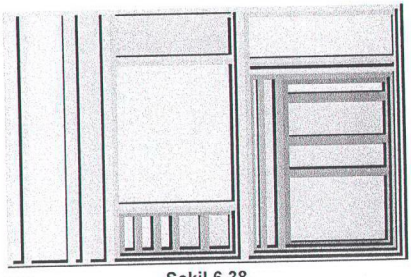
Top-down algoritmasıyla 640x480 çözünürlüklü bir monitörde 200 e kadar alt grup rahatlıkla gösterilebilinir. Şekil 6.37.'de bir şirketin organizasyon şemasının maaşlara göre Top-down algoritmasıyla görselleştirilmesi verilmiştir. Pozisyona göre alınan maaşlar açık ve koyu gri olarak belirtilmiştir.



Şekil 6.37:
Top-down algoritması
(Turo and Johnson,1992)

2. Slice and dice algoritması

Slice and dice algoritması, hem yatay hem de dikey olarak veri setini alt gruplara ayırıken yeniden bölümlendirme yapar. Böylelikle Top-down algoritmasına göre çok daha fazla alt grup görselleştirebilir. Bu sayı yaklaşık 1000 kadardır. Şekil 6.38'de aynı veri seti üzerinde Slice and dice algoritmasının uygulanması gösterilmektedir.



Şekil 6.38.
Slice and dice algoritması
(Turo and Johnson,1992)

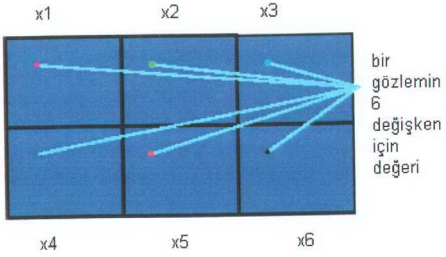
Treemap' in en büyük dezavantajı; gösterebildiği veri sayısının sınırlı olmasıdır. Scrolling ve panning teknikleri Treemap'in sınıra ulaşmasından sonra daha fazla veri gösterebilmesini sağlar fakat yapısının bozulmasına neden olabilir (Turo and Johnson, 1992).

Treemap'i geliştirmeyi amaçlayan çalışmalar halen devam etmektedir. Subdivision algoritmasını kullanarak ana parçayı, yapılan işlemin optimum olduğuna karar verdikten sonra bölümlere ayıran Squarified Treemap (Bruls, Huizing and Wijk, 2000) ve farklı bir uzay doldurma tekniği kullanan Cushion Treemap (Wijk and Wetering, 1999), son dönemde yapılan çalışmalarda geliştirilen Treemap örnekleridir. Şekil 6.39.'da bir Treemap örneği görülmektedir.

Şekil 6.39.
Treemap örneği
(Ng, 2000)

6.3.4. Piksel gösterim teknikleri

Piksel gösterim teknikleri, grafik alanı değişken sayısı kadar alt alanlara bölerek her bir gözlem değerini, ya da her bir gözlem değerinin önceden tanımlanmış olan ve veri tipine göre değişen fonksiyonlarla değişkenlere uzaklıklarını, belli aralıklar belli renklere denk gelecek şekilde renklendirerek 1 piksel boyutunda alt alan içinde görüntüler.

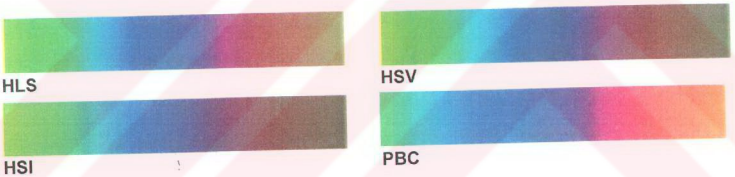


Şekil 6.40.
Bir gözlemin 6 değişken için değeri
(Keim, 1997)

Piksel gösterim teknikleri, diğer tekniklere göre çok daha fazla verinin (1000.000) üstüste yığılmadan tek bir grafik alanında görüntülenmesini sağlaması ve görsel olarak veri seti hakkında kolay algılanabilir bilgi verdiğinden diğer tekniklere göre daha fazla tercih edilirler.

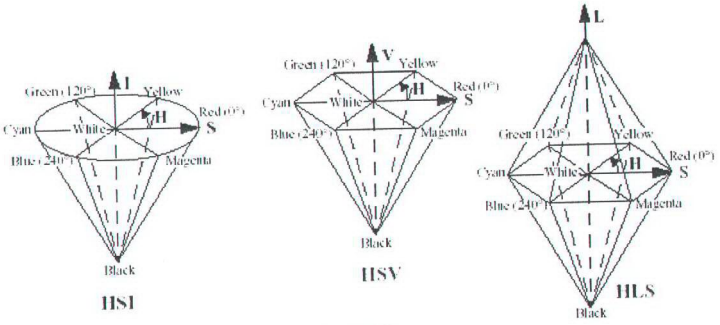
Renk atama

Gözlemlere renk atama işlemi, kısaca tek parametreli bir dağılımın seçilen renk modelliyle renklendirilmesidir. Renk modelleri; temel renk modeli olan PBC (Perception-based classification) renk modelinden lineer enterpolasyon kullanılarak türetilmişlerdir. HSI (hue, saturation, intensity), HSV ve HLS renk modellerine örnek olarak sayılabilir. Renk modellerinin kullandığı farklı renk spektrumları Şekil 6.41.'de gösterilmiştir.



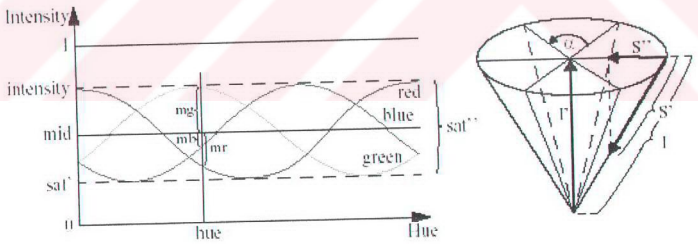
Şekil 6.41
Renk modelleri
(Ankerst, 2000)

Farklı renk modelleri, insan algısına yönelik yapılan çalışmalar sonucunda ortaya çıkmıştır. Her bir renk modeli, renk spektrumunu Şekil 6.42'de gösterilen farklı geometrik şekillere dayanarak oluşturur.



Şekil 6.42.
Renk modellerinin geometrisi
(Keim and Kriegel, 1995)

Renk modeline göre renklendirme yaparken renk modelinin geometrik şekline ait parametrele kullanılır. HSI modelinin geometrik şekli konidir. Koniye ait parametreler bir kosinünüs eğrisi üzerinde hesaplanır. Şekil 6.43. koniye ait parametreleri ve kosinüs eğrisini göstermektedir.



Şekil 6.43.
HSI renk modeli
(Keim and Kriegel, 1995)

HSI renk modelinde H renk; S doygunluk I da yoğunluk parametrelerini belirtmektedir. Kosinüs eğrisinden yararlanılarak hesaplanan bu parametreler aşağıdaki gibi formülize edilirler:

$$renk = a\cos\left(\frac{2xmr - mg - mb}{\sqrt{6}x\sqrt{mr^2 + mg^2 + mb^2}}\right) \quad (6.1)$$

mid bir sabit, doygunluk' da normalize edilmiş doygunluk olmak üzere;

$$doygunluk = 1 - \frac{doygunluk'}{yoğunluk} = \frac{2x(yoğunluk - mid)}{yoğunluk} \quad (6.2)$$

$$yoğunluk = mid + \sqrt{\frac{2}{3}x(mr^2 + mg^2 + mb^2)} \quad (6.3)$$

Ekrana yerleştirme

Değişkenlerin değerlerine göre farklı renkleri alan her bir gözlemin grafik alana yerleştirilmesinde farklı teknikler kullanılır. Bu teknikler veri seti üzerinde uygulanmak istenen amaca göre değişir. Veri seti üzerinde yapılan sorguların görselleştirilmesi amaçlandığında; sorguya bağımlı (query dependent) teknikler, verilerin görselleştirilmesi amaçlandığında ise sorgudan bağımsız (query indenpendent) teknikler kullanılır. (Keim and Kriegel, 1996)

Sorgudan bağımsız teknikler

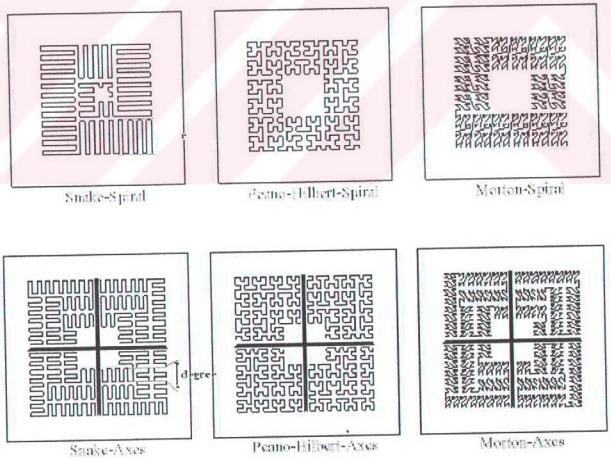
Sadece veri setinin grafiksel olarak görüntülenmesi amacı güdülüyorsa, verileri istenilen bir değişkene ya da değişkenlere göre sıralayan sorgudan bağımsız gösterim teknikleri kullanılır.

Sorgudan bağımsız gösterim teknikleri, piksellerin alanlara yerleştirilmesinde soldan sağa satır şeklinde, yukarıdan aşağı sütun şeklinde ya da uzay doldurma tekniklerini kullanır. Satır ve sütun yöntemleri, piksel genişliğinde yerleştirme yaptığından, görsel olarak çok tatmin edici sonuçlar vermeyebilir. Bunun yerine Peano, Hilbert ve Morton tarafından ortaya atılan farklı uzay doldurma eğrilerine dayalı ve iyi kümeleme gerçekleştirdiklerinden veri madenciliğine daha uygun olan tekrarlı örüntü (recursive pattern) teknikleri

kullanılır. Sorgudan bağımsız gösterim teknikleri özellikle zaman serileri gibi tek bir değişkeni doğal olarak sıralanmış veri setlerinde oldukça kullanışlıdır.

Sorguya dayalı teknikler

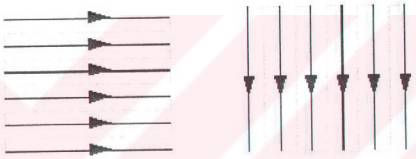
Sorguya dayalı teknikler, tüm veri setini değil, sadece belli bir sorguyla araştırılan özelliklere sahip olan verileri görüntüler. Sorgudan bağımsız teknikler gibi gözlemlere doğrudan renkli bir piksel atamak yerine tanımlı bazı uzaklık kavramlarını kullanarak renklendirilirler. Bu uzaklıklar; veriler ile sorgulanan değerlerler arasında bir uzaklık olup veri türüne ve yapılan uygulamaya göre değişen bir uzaklık fonksiyonuyla belirlenir. Genellikle tüm değişkenler aynı anda sorgulanmadığından, sorgulanmayan değişkenlerin grafikte etkisi yoktur. Şekil 6.44. farklı uzay doldurma tekniklerini göstermektedir.



Şekil 6.44.
Uzay doldurma teknikleri
(Keim and Kriegel, 1996)

6.3.4.1. Tekrarlı Örüntü (Recursive Pattern)

Tekrarlı Örüntü, sorgudan bağımsız gösterim tekniklerindendir. Verilerin satırlar şeklinde yanyana ya da sütunlar şeklinde altalta grafik alana yerleştirilmesi en basit tekrarlı örüntü tekniğidir. (Şekil 6.45) Fakat 1 veri için grafik alanda 1 piksel yer ayrılması ve fazla sayıda verinin olması nedeniyle bu teknik doyurucu sonuç vermez. Bunun için pikseller ekrana küçük gruplar oluşturacak ve bir örüntüyü gösterecek şekilde tasarlanarak yerleştirilirler. Bu yerleştirme işlemi, genel tekrar şemasına dayalı yerleştirme olarak da bilinir.



Şekil 6.45.
Verilerin satır ve sütun olarak yerleşme şeması
(Keim, Kriegel and Ankerst, 1995)

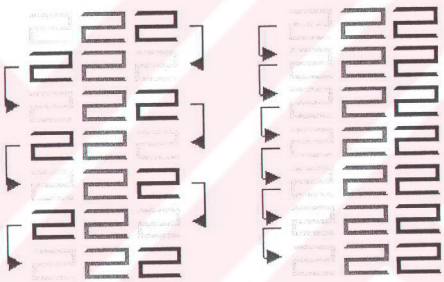
Piksellerin grafik alana yerleştirme işlemi 2 adımda tamamlanır. İlk adımda piksellerin gruplanmasıyla oluşacak olan örüntüye (birinci örüntü) karar verilir. Daha sonraki adımda ise ikinci örüntüyü oluşturacak düzenleme şekli seçilir. Düzenleme sonucu elde edilen yapı, üçüncü bir örüntü için temel alınabilir. Örüntü sayısı genel recursive şemasını oluşturur ve keyfi olarak seçilir (Keim, Kriegel and Ankerst, 1995). Örnek olarak bir zaman serisi ele alındığında, birinci örüntü günlük, ikinci örüntü haftalık, üçüncü örüntü de yıllık veriler olarak alınabilir.

Tekrarlı Örüntü, 'geri ve ileri' düzenlemeye dayanır. Belirlenen sayıda veriler; önce soldan sağa, sonra aşağıya, aşağıdan sola ve tekrar bulunduğu yerden sağa dönecek şekilde bir şema halinde yerleştirilirler. Bu kadarlık alan şemanın 1 düzeyi olarak ifade edilir. Bu yerleştirme işleminin farklı uygulamaları Şekil 6.46.' ve Şekil 6.47.'de gösterilmiştir.

Şekil 6.46.
Satır yerleştirme

Şekil 6.47.
Sütun yerleştirme

Veriler kullanılan şemaya göre yerleştirilirken bir düzey, kendisinden önceki düzeyde oluşan örüntüyü kullanır. Şekil 6.48. farklı düzenlemiş şemaları göstermektedir.

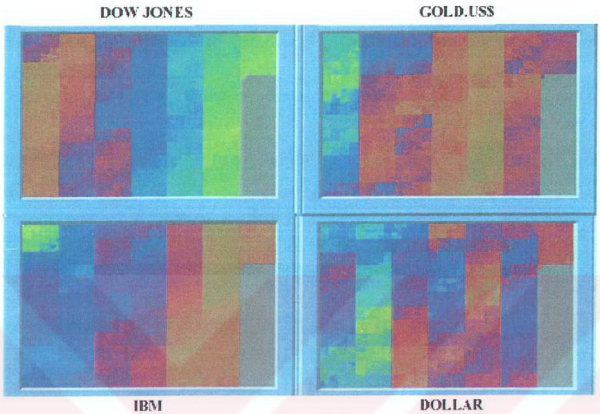


Şekil 6.48.
Farklı şemalar

i. düzeyde w_i yatay olarak yerleştirilen veri sayısı, h_i; i. düzeyde satır sayısı olmak üzere; i. düzeyde şema içine yerleştirilecek veri sayısı $w_i x h_i$ olarak ifade edilir. k seviyeli bir şemada gösterilebilecek maksimum veri sayısı ise $\prod_{i=1}^{k} w_i x h_i$ kadardır. w_i ve h_i parametrelerinin kullanıcı tarafından belirlenmesi farklı grafiklerin elde edilmesine neden olur.

Şekil 6.49.'da 4 değişkenli bir zaman serisinin tekrarlı örüntü ile görselleştirilmesi verilmiştir. Grafik alandaki 7 sütun, 3. düzey örüntü olup yıllara, 12 satır ise 2. düzey örüntü olup aylara denk düşmektedir. Açık renkli

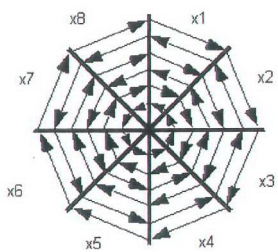
pikseller verilerin yüksek değerleri için kullanılmıştır. Verilerin değerleri düştükçe renkleri de koyulaşmaktadır.



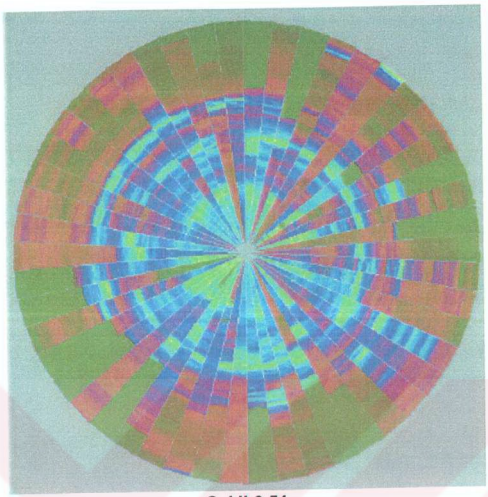
Şekil 6.49.
Tekrarlı Örüntü
(Ankerst, 2001)

6.3.4.2. Dairesel Bölümler (Circle Segments)

Dairesel Bölümler, Tekrarlı Örüntü'nün şemalarını kullanarak, veri setini daire şeklindeki bir grafik alan içinde görselleştirir. Veri setinde k tane değişken varsa, daireyi k eşit parçaya böler. Her dilimi bir değişkene denk düşer Dairesel Bölümler, veriler dairenin merkezinden dış çeperine doğru 'geri ve ileri' düzenlemeyle yerleştirir (Ankerst, Keim and Kriedel, 1996).



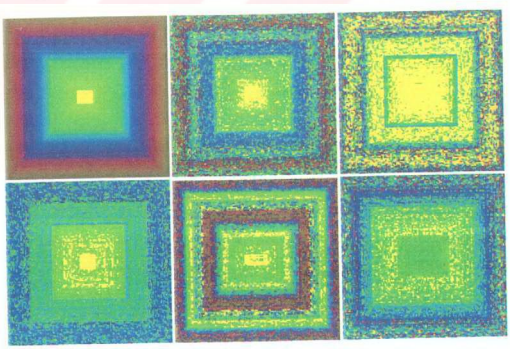
Şekil 6.50.
8 değişkenli Dairesel Bölümler



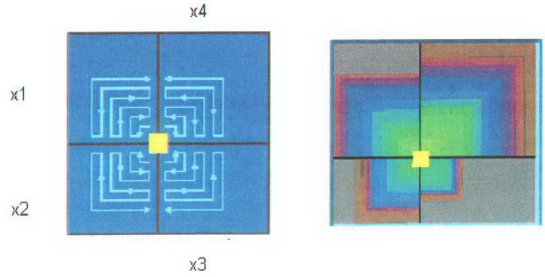
Şekil 6.51
Dairesel Bölümler
(Ankearst, 2001)

6.3.4.3. Spiral ve Eksen teknikleri

Spiral ve Eksen gösterim teknikleri, sorguya bağımlı tekniklerdir. Grafik alanı sorgulanan değişkenler kadar parçaya ayırarak tanımlı olan uzaklık fonksiyonlarına göre verileri yerleştirirler.



Şekil 6.52
Spiral Tekniği



Şekil 6.53
Eksen tekniği

6.3.5. Diğer gösterim teknikleri

Grafik esaslı ve karma gösterim teknikleri özel durumlarla ilgilendiklerinden burada söz edilmemektedir.

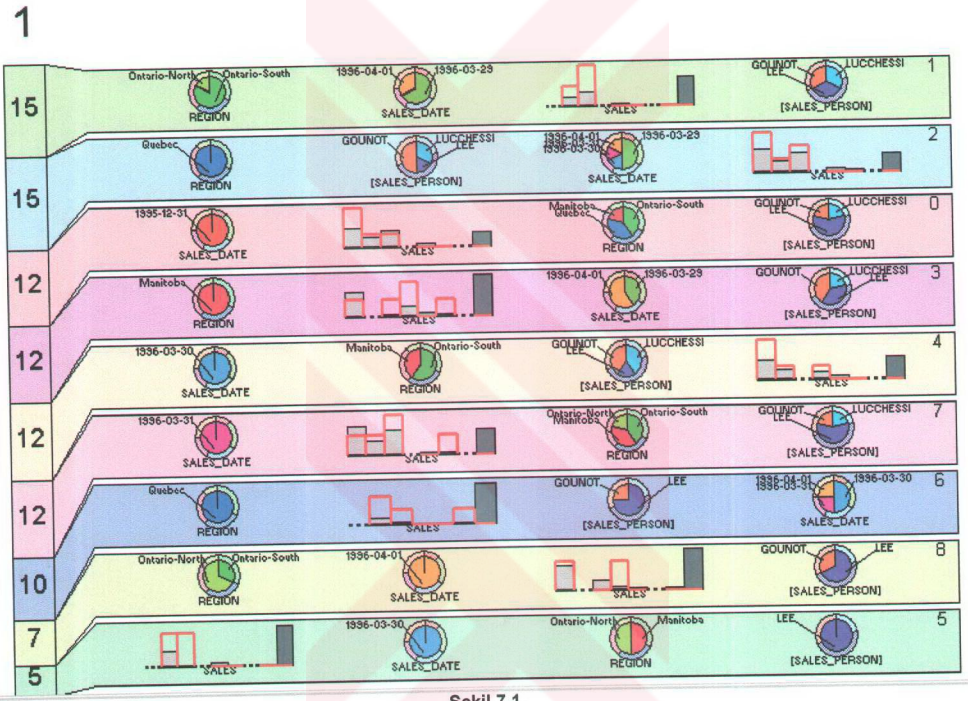
YEDİNCİ BÖLÜM
UYGULAMA

Veri mandenciliği teknikleri uygulaması için belli başlı veri madenciliği tprogramlarından biri olan IBM Intelligent Miner kullanılmıştır. 11 tablodan ve yaklaşık olarak 35 alandan oluşan bir şirketin satış deopartmanına ait ilişkisel bir veritabanı üzerinde programda varolan tekniklerden kümeleme, yapay sinir ağları, karar ağaçları kullanılarak sınıflandırma ve Treemap uygulaması yapımıştır. Program yapılan analizlerin görsel olarak desteklenmesine genetik algoritmaların uygulanmasına olanak sağlamamıştır.

Programın kendi kümeleme algoritması kullanılmış, ve maksimum küme sayısı 9 olarak belirlenmiştir.

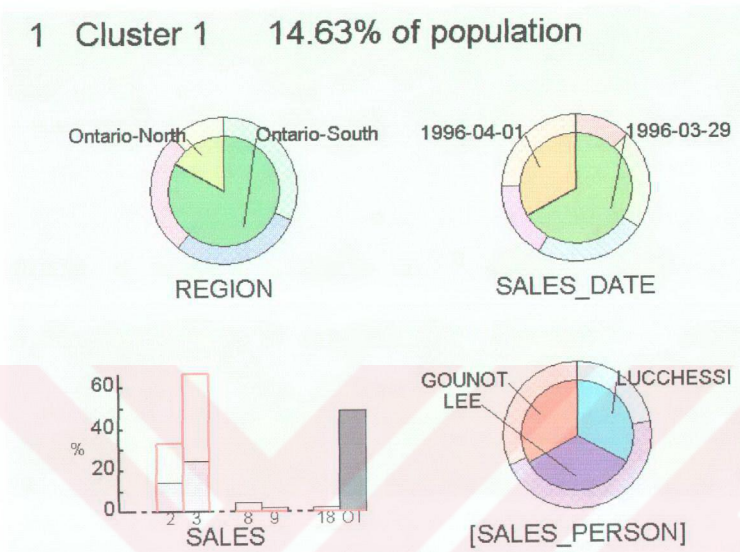
Şekil 7.1.'den görülebileceği gibi kümeleme algoritması 9 farklı küme oluşturmuştur. Kümeler sırasıyla veri setinin %15, %15, %12, %12, %12, %12, %10, %7 ve %5' i kadar veriyi içerirler.

Şekil 7.1'deki satırlar, kümelere denk gelir. Kümeler içindeki grafikler ise sırasıyla kümenin oluşmasında etkili olan değişkenleri gösterir.



Şekil 7.1.
IBM IM kümeleme sonuçları

Birinci küme



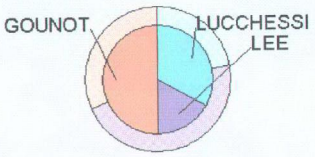
Birinci kümeyi oluşturan en büyük değişken satışların olduğu bölgeleri gösteren değişkendir. Sayısal veri olmadığında, bölge değişkeni daire şeklinde görselleştirilmiştir. Dairenin iç kısmı küme içindeki dağılımı gösterirken dış taraftaki dar halka tüm veritabanındaki dağılımı göstermektedir. Oluşturulan kümeler ve veri seti arsında kıyaslama yapabilmek için böyle bir yola başvurulmuştur.

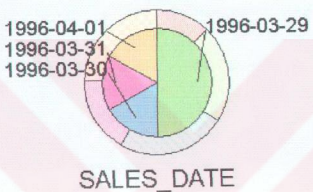
İkinci en büyük isedeğişken satış zamanlarını gösteren değişkendir. Bu değişken de sayısal veri içermediğinden daire biçiminde görselleştirilmiştir.

Birinci kümeyi oluşturan en büyük üçüncü değişken ise satışları gösteren değişkendir. Bu değişken sayısal veri içerdiğinden çubuk grafikle görselleştirlmiştir. Kırmızı sınırlar küme içindeki satışların tüm veri setindeki satışlarla kıyaslanması için kullanılmıştır.

İkinci Küme

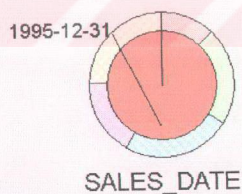
1 Cluster 2 14.63% of population

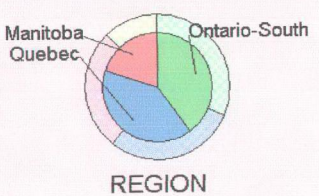


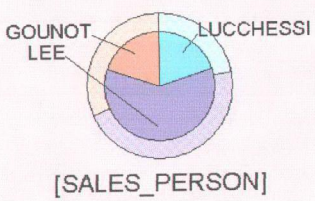


Üçüncü Küme

1 Cluster 0 12.20% of population

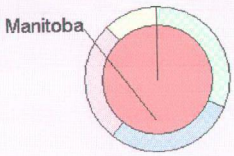




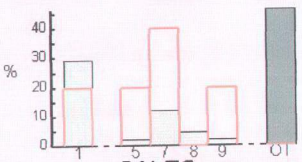


Dördüncü küme

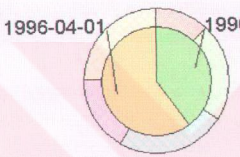
1 Cluster 3 12.20% of population



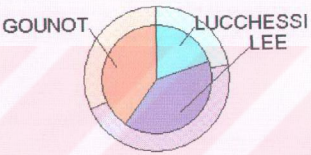
REGION



SALES



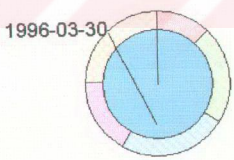
SALES_DATE



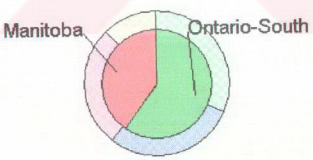
[SALES_PERSON]

Beşinci Küme

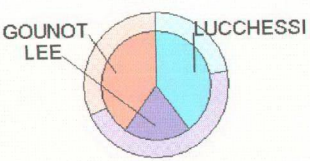
1 Cluster 4 12.20% of population



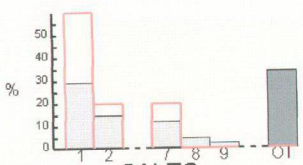
SALES_DATE



REGION



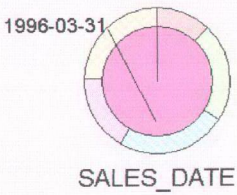
[SALES_PERSON]

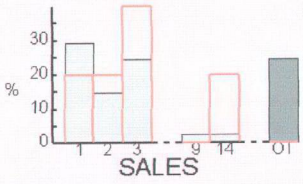


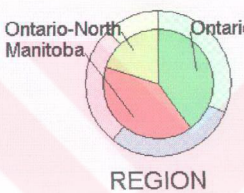
SALES

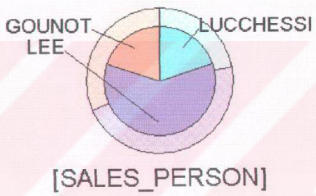
Altıncı küme

1 Cluster 7 12.20% of population



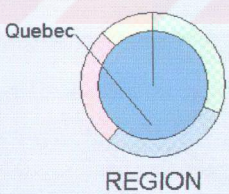


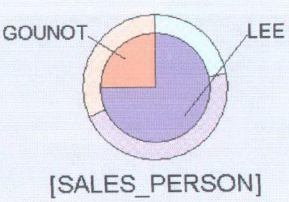


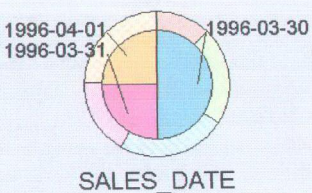


Yedinci Küme

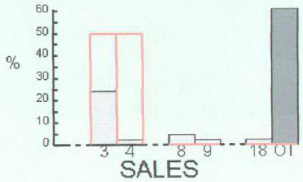
1 Cluster 6 9.76% of population



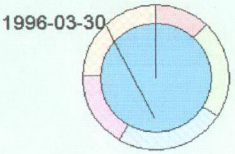




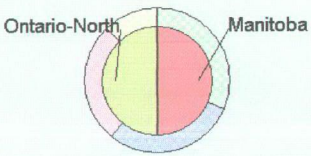
1 Cluster 5 4.88% of population



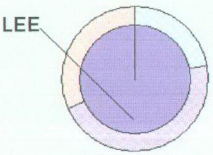
SALES



SALES_DATE



REGION

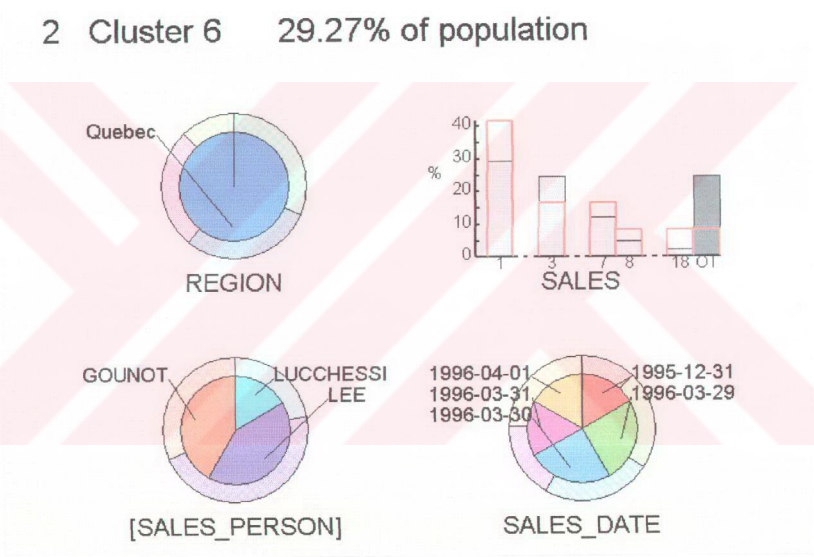


[SALES_PERSON]

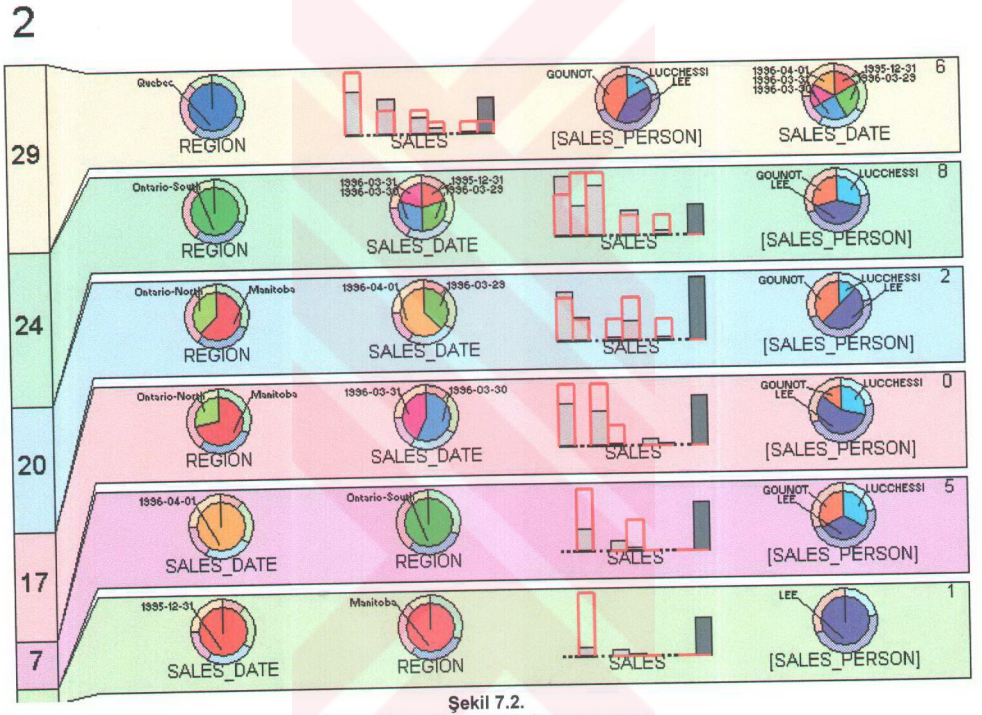
Şekil 7.2. ' de IBM IM' nin kullandığı yapay sinir ağı algoritmasıyla veri setinin kümelere ayrılmasını gösterilmiştir. Oluşturulacak maksimum küme sayısı 9 olarak belirlenmiştir, fakat algoritma 6 küme oluşturabilmiştir.

Kümeler sırasıyla veri setinin %29, %24, %20, %17 ve %7'si kadar veri içerir.

Birinci küme



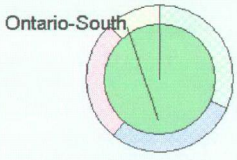
Yapay sinir ağlarıyla oluşturulan ilk küme veri setinin %29'unu oluşturur. Bu kümeyi oluşturan en büyük değişken satış bölgelerinin gösteren değişkendir. Küme tek bir bölgeden oluşmaktadır. Kümeyi oluşturan ikinci büyük değişken satış miktarlarını gösteren değişkendir. Kümeyi oluşturan diğer değişkenler sırasıyla satışları yapan kişiler ve satış zamanlarıdır.
Diğer kümeler de benzer şekilde gösterilmiştir.



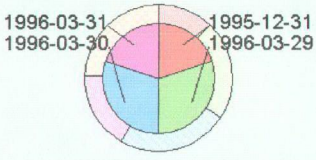
Şekil 7.2.
Yapay sinir ağlarıyla kümeleme şeması

İkinci Küme

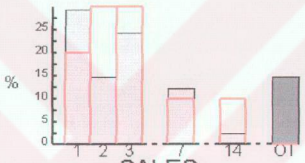
2 Cluster 8 24.39% of population



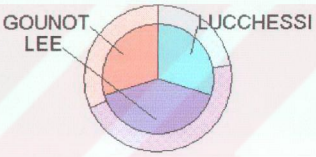
REGION



SALES_DATE



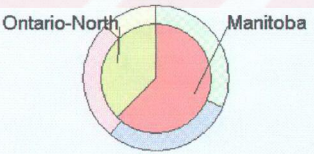
SALES



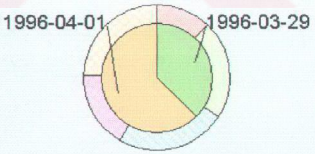
[SALES_PERSON]

Üçüncü Küme

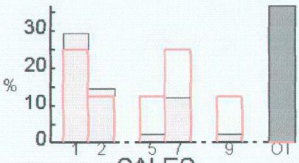
2 Cluster 2 19.51% of population



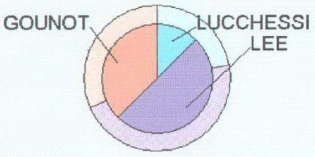
REGION



SALES_DATE



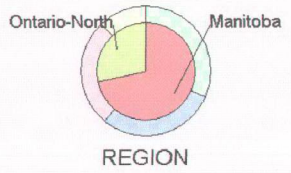
SALES

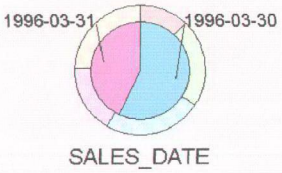


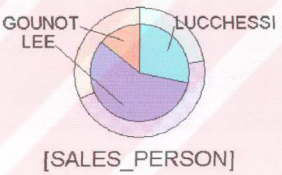
[SALES_PERSON]

Dördüncü Küme

2 Cluster 0 17.07% of population

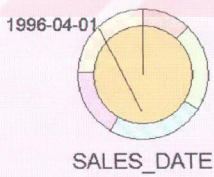


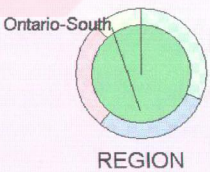


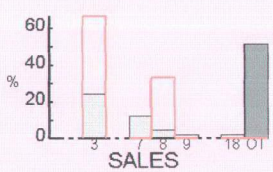


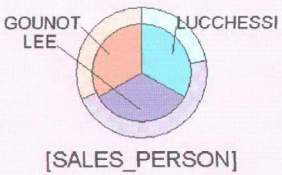
Beşinci Küme

2 Cluster 5 7.32% of population

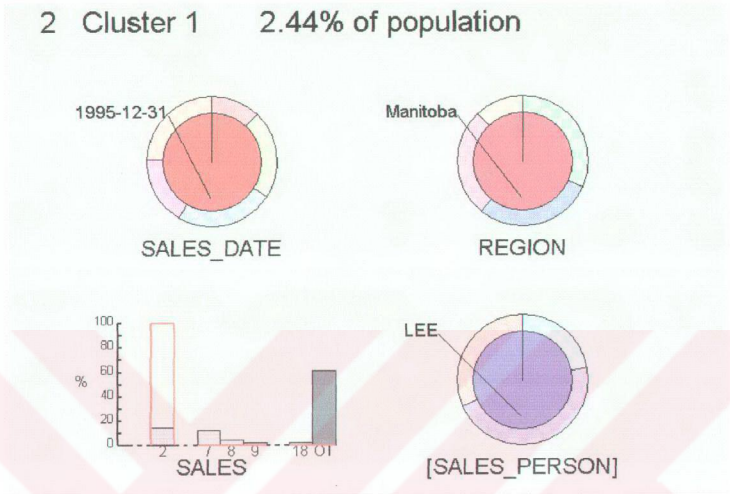




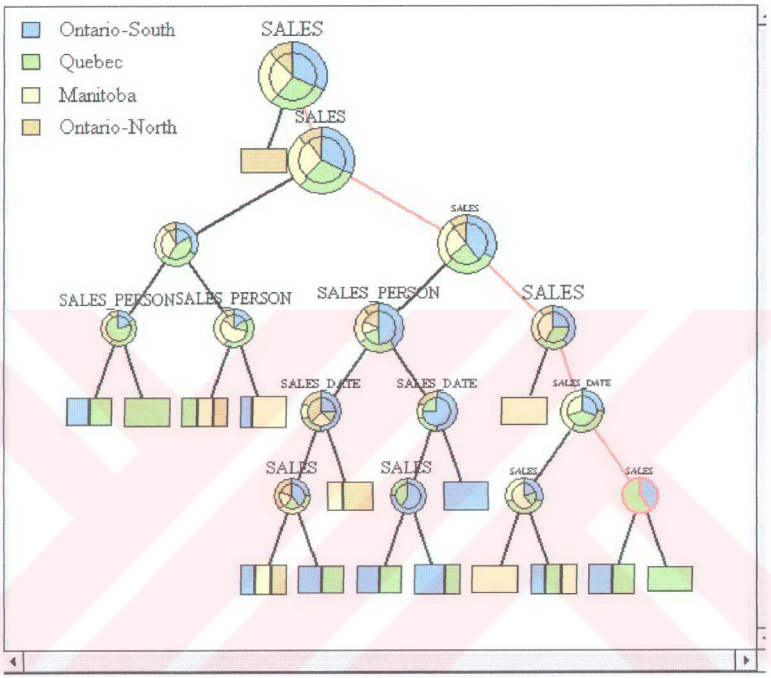


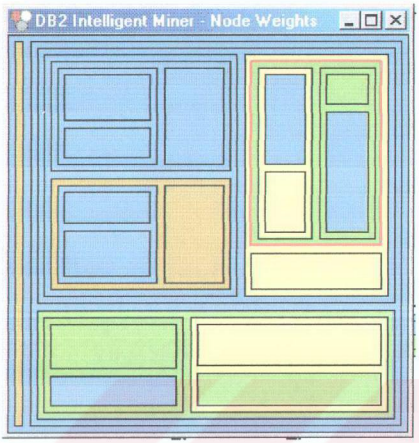


Altıncı Küme



Satışlar üzerinde karar ağacının uygulanması





Treemap

KAYNAKLAR

ADRİAANS, P and ZANTİNGE D., Data Mining , 1996, Addison-Wesley

ANKERST M., KEIM D. A., and KRIEGEL, H.-P., 1996, "Circle segments: A technique for visually exploring large multidimensional data sets," in Proc. Visualization 96, Hot Topic Session, San Francisco, CA

ANKERST, M., 2000, Visual Data Mining, Dissertation im Fach Informatik an der Fakultät für Mathematik und Informatik der Ludwig-Maximilians-Universität, München

BARBARIA, K., 2001, "Introduction to Treemap", tutorial, http://www.cs.umd.edu/hcil/treemap3/tutorial.html

BERSON, A., Kurt THEARLING, K., and SMITH, S., 1999, Building Data Mining Applications for CRM, McGraw-Hill

BRULS, M., HUIZING, K. and WIJK, J. J., 2000, " Squarified treemaps", In Proceedings of the Joint Eurographics and IEEE TCVG Symposium on Visualization, pages 33—42

BRUNSDIN, C., FOTHERINGHAM, A. S. and CHARTLTION, M. E., "An Investigation of Methods for Visualising Highly Multivariate Datasets" , The Advisory Group on Computer Graphics (AGOCG) Technical Report

BUJA, A., SWAYNE, D. F. and COOK, D., 1996, "Interactive High-Dimensional Data Visualization", Journal of Computational and Graphical Statistics, Vol. 5, No. 1, pp. 78-99

FASULO, D., 26 April 1999, An Analysis Of Recent Work on Clustering Algorithms, Technical Report no 01-03-02, Department of Computer and Science & Engineering, University of Washington

FEINER S. and BESHERS C., 1990 "Visualizing n-Dimensional Virtual Worlds with n-Vision", Computer Graphics, Vol. 24, No. 2, pp. 37-38

FEINER S. and BESHERS C., 1990, "Worlds within Worlds: Metaphors for Exploring n-dimensional Virtual Worlds", Proc. UIST, pp. 76-83

FRALEY, C., and RAFTERY , A. E., 27 february 1998, How Many Clusters? Which Clustering Method? Answers Via Model_Based Cluster

Analysis , Technical Report no.329, Department of Statistics; University of Washington.

FURNAS, G. W. and BUJA A., 1994 "Prosections views: Dimensional inference through sections and projections," Journal of Computational and Graphical Statistics, vol. 3, no. 4, pp. 323–353

GRAEPEL, T., Statistical Physics of Clustering Algorithms, Diplomarbeit, Technische Universitat Berlin, April 1998 tez

GUPTA, S. K. and SABBASIVA RAOK, K. and Bhatnagar, V., September 1999, "K-means Clustering Algorithms for Categorical Attributes", DaWaK99 (Data Warehouse and Knowledge Discovery), Florence, Italy

NG, K.-C., Interactive Visualisation Techniques For Ontology Developmenta Thesis Submitted To The University Of Manchester For The Degree Of Doctor Of Philosophy In The Faculty Of Science And Engineering, November 2000

HEARST, M., 1995, "Tilebars: Visualization of term distribution information in full text information access," in Proc. of ACM Human Factors in Computing Systems Conf. (CHI'95), pp. 59–66

HEASRT, M.A., May 1995 "TileBars: Visualization of Term Distribution Information in Full Text Infırmation Acces", Prooceedings of CHI'95, Denver,VCO

KASKI, S., NIKKILA, J. and KOHONEN, T., "Methods for Exploratory Cluster Analysis", International Conferance on Advances in Infrastructure for Electronic Business, Science, and Education on Internet, L'Aquila, July 31-August 6. Romoli, 2000. (proceedings on CD-ROM)

KEIM, D.A., and KRIEGEL H.-P., 1995, "Issues in Visualizing Large Databases" Visual Database Systems, pp.203-214, Chapman & Hall Lrd

KEIM, D. A. and KRIEGEL, H.-P., 1995, "Issues in Visualizing Large Databases" Published in: Proc. Int. Conf. on Visual Database Systems (VDB-3), Lausanne, Schweiz, März 1995, in: Visual Database Systems, Chapman & Hall Ltd.

KEIM, D. A. , KRIEGEL H.-P.and ANKERST, M., 1995, "Recursive Pattern: A Technique for Visualizing Very Large Amounts of Data", Proc. Visualization '95, Atlanta, GA, pp. 279-286

KEIM, D A. and KRIEGEL,H.-P., 6 December 1996, "Visualization Techniques for Mining Large Databases: A Comparison", Transactions on Knowledge and Data Engineering, Vol. 8, No., pp. 923-938

KEIM, D. A., 1996, "Pixel-oriented Visualization Techniques for Exploring Very Large Databases", Journal of Computational and Graphical Statistics, Vol. 5, No. 1, pp. 58-77.

KEIM, D.A., 1997, "Visual Techniques for Exploring Databases", tutorial, Conf. on Very Large Databases (VLDB'97), Athens, Greece

KEIM, D. A., "Information Visualization And Visual Data Mining" IEEE Transactions On Visualization And Computer Graphics, Vol. Xx, No. Yy, Month 2001

MCLEOD, A. I. and PROVOST, S.B., January 26, 2001, Multivariate Data Visualization

NIGGEMANN, O., 2001, Visual Data Mining of Graph-Based Data, Department of Mathematics and Computer Science, University of Paderborn, Germany , Dissertation

PICKETT, R. M. and GRINSTEIN, G. G., 1988, "Iconographic displays for visualizing multidimensional data," in Proc. IEEE Conf. On Systems, Man and Cybernetics, IEEE Press, Piscataway, NJ, pp. 514–519

ROBERSTSON, G. , MACKINLAY, J. D. and CARD, S. K., 1991, "Cone Trees: Animated 3D Visualizations of Hierarchical Information", Proc. Human Factors in Computing Systems CHI '91 Conf., New Orleans, LA, pp. 189-194

TURO, D. and JOHNSON, B., October 1992, "Improving the Visualization Of Hierarchies with Treemaps: Design Issues and Experimentation", Proceedings of the IEEE Conferance on Visualization

UNWIN, A., VOLINSKY, C. and WINKLER, S. " Parallel Coordinates for Exploratory Modelling Analysis" Submitted To University at Albany, Center For Social and Demographic analysis, Working papers, University at Albany, 2000

UNWIN, A., 2000, "Visualisation for Data Mining", International Conference on Data Mining, Visualization and Statistical System, Seoul

WEGMAN, E. J. and CARR, D. B., Statistical Graphics and Visualization, center for Computational Statistics George Mason University, e-book, 2001

WIJK, J. J., and LIERE, R. D., 1993, "Hyperslice," in Proc. Visualization '93, San Jose, CA, pp. 119–125

WONG, P. C. and BERGERON, R. D., 1995, "30 Years of Multidimensional Multivariate Visualization", Proc. Workshop on Scientific Visualization, IEEE Computer Society Press

ZAIANE, O. R. and Foss, A. and Lee, C. H. and WANG, W., May, 2002, On Data Clustering Analysis: Scalability, Constraints and Validation, Proc. of the Sixth Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'02), Taipei, Taiwan, pp 28-39