**Data Warehouse: Basic Concepts**

**#Data Warehouse;**

Pek çok farklı şekilde tanımlanmış, ancak kesin olarak değil.

Kuruluşun operasyonel veri tabanından ayrı tutulan bir karar destek veri tabanı

Analiz için birleştirilmiş, geçmiş verilerden oluşan sağlam bir platform sağlayarak bilgi işlemeyi destekleyin.

“Veri ambarı, yönetimin karar verme sürecini desteklemek için konuya yönelik, entegre, zamana bağlı ve geçici olmayan bir veri koleksiyonudur.

**#Data warehousing:**

Veri ambarlarını oluşturma ve kullanma süreci

Müşteri, ürün, satış gibi ana başlıklar etrafında organize edilmiş

Günlük operasyonlara veya işlem işlemeye değil, karar vericiler için verilerin modellenmesine ve analizine odaklanmak

Karar destek sürecinde yararlı olmayan verileri hariç tutarak belirli konu sorunları hakkında basit ve özlü bir görüş sağlayın

**#Data Warehouse—Integrated;**

Çoklu, heterojen veri kaynaklarının entegre edilmesiyle oluşturulmuştur

ilişkisel veritabanları, düz dosyalar, çevrimiçi işlem kayıtları

Veri temizleme ve veri entegrasyon teknikleri uygulanır.

Farklı veri kaynakları arasında adlandırma kurallarında, kodlama yapılarında, öznitelik ölçülerinde vb. tutarlılık sağlayın

Örneğin, Otel fiyatı: para birimi, vergi, kahvaltı dahil vb.

Veriler ambara taşındığında dönüştürülür.

**#Data Warehouse—Time Variant**

Veri ambarı için zaman ufku, operasyonel sistemlerinkinden önemli ölçüde daha uzundur

Operasyonel veritabanı: geçerli değer verileri

Veri ambarı verileri: tarihsel bir perspektiften bilgi sağlar (ör. son 5-10 yıl)

Veri ambarındaki her anahtar yapı

Açıkça veya dolaylı olarak bir zaman öğesi içerir

Ancak operasyonel verilerin anahtarı “zaman unsuru” içerebilir veya içermeyebilir.

**#Data Warehouse—Nonvolatile**

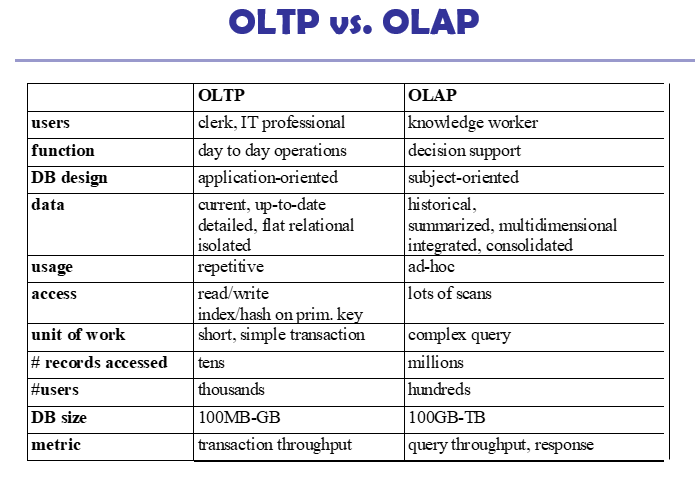
Operasyonel ortamdan dönüştürülen fiziksel olarak ayrı bir veri deposu

Veri ambarı ortamında operasyonel veri güncellemesi gerçekleşmez

İşlem işleme, kurtarma ve eşzamanlılık kontrol mekanizmaları gerektirmez

Veri erişiminde yalnızca iki işlem gerektirir:

verilerin ilk yüklenmesi ve verilere erişim



**#Why a Separate Data Warehouse?**

Her iki sistem için yüksek performans

DBMS— OLTP için ayarlanmış: erişim yöntemleri, indeksleme, eşzamanlılık kontrolü, kurtarma

Ambar—OLAP için ayarlanmış: karmaşık OLAP sorguları, çok boyutlu görünüm, konsolidasyon

Farklı işlevler ve farklı veriler:

eksik veriler: Karar desteği, operasyonel veritabanlarının tipik olarak korumadığı geçmiş verileri gerektirir

veri konsolidasyonu: DS, heterojen kaynaklardan gelen verilerin konsolidasyonunu (toplama, özetleme) gerektirir

veri kalitesi: farklı kaynaklar genellikle uzlaştırılması gereken tutarsız veri temsilleri, kodlar ve biçimler kullanır

Not: OLAP analizini doğrudan ilişkisel veritabanlarında gerçekleştiren daha fazla sistem vardır.

#Three Data Warehouse Models

kurumsal depo, Enterprise warehouse

tüm organizasyonu kapsayan konularla ilgili tüm bilgileri toplar

Veri Pazarı, Data Mart

belirli bir kullanıcı grubu için değer taşıyan şirket genelindeki verilerin bir alt kümesi. Kapsamı, pazarlama verileri pazarı gibi belirli, seçilmiş gruplarla sınırlıdır.

Bağımsız ve bağımlı (doğrudan depodan) veri pazarı

Sanal depo, Virtual warehouse

Operasyonel veritabanları üzerinde bir dizi görünüm

Muhtemel özet görünümlerin sadece bir kısmı gerçekleştirilebilir.

#Extraction, Transformation, and Loading (ETL)

Veri çıkarma, Data extraction

çoklu, heterojen ve harici kaynaklardan veri alma

Veri temizleme, Data cleaning

verilerdeki hataları tespit edin ve mümkün olduğunda düzeltin

Veri dönüşümü, Data transformation

verileri eski veya ana bilgisayar biçiminden ambar biçimine dönüştürün

Yük, Load

görünümleri sıralayın, özetleyin, birleştirin, hesaplayın, bütünlüğü kontrol edin ve indeksler ve bölümler oluşturun

Yenile, Refresh

güncellemeleri veri kaynaklarından ambara yaymak

**#Metadata Repository**

Meta veriler, ambar nesnelerini tanımlayan verilerdir. Depolar:

Veri ambarı yapısının açıklaması

şema, görünüm, boyutlar, hiyerarşiler, türetilmiş veri tanımı, veri pazarı konumları ve içerikleri

Operasyonel meta veri

veri kökeni (geçirilen verilerin ve dönüştürme yolunun geçmişi), verilerin para birimi (etkin, arşivlenmiş veya temizlenmiş), izleme bilgileri (depo kullanım istatistikleri, hata raporları, denetim izleri)

Özetleme için kullanılan algoritmalar

Operasyonel ortamdan veri ambarına eşleme

Sistem performansı ile ilgili veriler

ambar şeması, görünüm ve türetilmiş veri tanımları

İş verisi

iş terimleri ve tanımları, veri sahipliği, ücretlendirme politikaları

**#Data Warehouse Modeling: Data Cube and OLAP**

Bir veri ambarı, verileri bir veri küpü biçiminde görüntüleyen çok boyutlu bir veri modeline dayanır.

Satış gibi bir veri küpü, verilerin birden çok boyutta modellenmesine ve görüntülenmesine olanak tanır.

Öğe (öğe\_adı, marka, tür) veya saat (gün, hafta, ay, çeyrek, yıl) gibi boyut tabloları

Olgu tablosu, ölçümleri (dolar\_sold gibi) ve ilgili boyut tablolarının her birinin anahtarlarını içerir.

Veri ambarı literatüründe, bir n-D taban küpü, temel küboid olarak adlandırılır. En yüksek özetleme düzeyine sahip olan en üstteki 0 boyutlu küboid, tepe küboid olarak adlandırılır. Küboidlerin kafesi bir veri küpü oluşturur.

#Conceptual Modeling of Data Warehouses

Veri ambarlarının modellenmesi: boyutlar ve ölçüler

Yıldız şeması, Star schema: Ortada bir dizi boyut tablosuna bağlı bir olgu tablosu

Kar tanesi şeması, Snowflake schema: Bazı boyutsal hiyerarşinin bir dizi daha küçük boyut tablosuna normalleştirildiği ve kar tanesine benzer bir şekil oluşturduğu bir yıldız şeması iyileştirmesi

Olgu takımyıldızları, Fact constellations: Çoklu olgu tabloları, bir yıldız koleksiyonu olarak görülen boyut tablolarını paylaşır, bu nedenle galaksi şeması veya olgu takımyıldızı olarak adlandırılır.

#**Data Cube Measures**: Three Categories

Dağıtıcı, Distributive: işlevi n toplam değere uygulayarak elde edilen sonuç, işlevi bölümleme olmadan tüm verilere uygulayarak elde edilen sonuçla aynıysa

Örn., say(), toplam(), min(), maks()

Cebirsel, Algebraic: her biri bir dağıtıcı toplama işlevi uygulanarak elde edilen M bağımsız değişkenli (burada M sınırlı bir tam sayıdır) cebirsel bir işlevle hesaplanabiliyorsa

Örn. avg(), min\_N(), standart\_sapma()

Bütünsel, Holistic: bir alt kümeyi tanımlamak için gereken depolama boyutunda sabit bir sınır yoksa.

Örn. medyan(), mod(), sıralama()

**#Typical OLAP Operations**

Topla (detay), Roll up (drill-up): verileri özetle

hiyerarşiye tırmanarak veya boyut küçülterek

Detaya inme (yuvarlanma), Drill down (roll down): toplamanın tersi

üst düzey özetten alt düzey özete veya ayrıntılı verilere veya yeni boyutların tanıtılmasına

Dilimleyin ve zar atın: projelendirin ve seçin

Döndür (döndür), Pivot (rotate):

küpü yeniden yönlendirme, görselleştirme, 3B'den 2B düzlem serisine

Diğer işlemler

detaya inme, drill across: birden fazla olgu tablosunu (çapraz) dahil etme

detaya inme, drill through: küpün alt seviyesinden arka uç ilişkisel tablolarına (SQL kullanarak)

8.5

**#Model Evaluation and Selection**

Değerlendirme ölçütleri: Doğruluğu nasıl ölçebiliriz? Dikkate alınması gereken diğer ölçütler?

Doğruluğu değerlendirirken eğitim seti yerine sınıf etiketli demetlerden oluşan doğrulama testi seti kullanın

Bir sınıflandırıcının doğruluğunu tahmin etme yöntemleri:

Bekletme yöntemi, rastgele alt örnekleme

Çapraz doğrulama

önyükleme

Sınıflandırıcıların karşılaştırılması:

Güvenilirlik aralığı

Maliyet-fayda analizi ve ROC Eğrileri

#**Evaluating Classifier Accuracy: Holdout & Cross-Validation Methods**

bekletme yöntemi

Verilen veriler rastgele iki bağımsız kümeye bölünür

Model yapımı için eğitim seti (ör. 2/3)

Doğruluk tahmini için test seti (örn. 1/3)

Rastgele örnekleme: uzatmanın bir varyasyonu

Uzatmayı k kez tekrarlayın, doğruluk = ort. elde edilen doğrulukların

Çapraz doğrulama (k-katlama, burada k = 10 en popülerdir)

Verileri, her biri yaklaşık olarak eşit boyutta olan, birbirini dışlayan k alt kümeye rasgele bölün

i. yinelemede, test seti olarak Di'yi ve eğitim seti olarak diğerlerini kullanın

Birini dışarıda bırakma: küçük boyutlu veriler için k = demet sayısı olduğunda k kıvrım

\*Tabakalı çapraz doğrulama\*: kıvrımlar, sınıf dist. her katlamada yakl. ilk verilerdeki ile aynı

#**Evaluating Classifier Accuracy: Bootstrap**

önyükleme

Küçük veri kümeleriyle iyi çalışır

Verilen eğitim demetlerini değiştirme ile aynı şekilde örnekler

yani, bir demet her seçildiğinde, yeniden seçilme ve eğitim setine yeniden eklenme olasılığı eşittir.

Birkaç önyükleme yöntemi ve yaygın olanı .632 boostrap'tir.

d demetli bir veri seti, d kez örneklenir ve değiştirilerek d numunelik bir eğitim seti elde edilir. Eğitim setine girmeyen veri grupları sonunda test setini oluşturur. Orijinal verilerin yaklaşık %63,2'si önyüklemede son bulur ve geri kalan %36,8'i test setini oluşturur (çünkü (1 – 1/d)d ≈ e-1 = 0,368)

Örnekleme prosedürünü k kez tekrarlayın, modelin genel doğruluğu:

#**Estimating Confidence Intervals: Classifier Models M1 vs. M2**

M1 ve M2 olmak üzere 2 sınıflandırıcımız olduğunu varsayalım, hangisi daha iyi?

elde etmek için 10 kat çapraz doğrulama kullanın ve

Bu ortalama hata oranları, yalnızca gelecekteki veri vakalarının gerçek popülasyonundaki hata tahminleridir.

Ya 2 hata oranı arasındaki fark sadece şansa atfedilirse?

İstatistiksel anlamlılık testi kullanın

Hata tahminlerimiz için güven sınırları elde edin

#**Estimating Confidence Intervals: Null Hypothesis**

10 kat çapraz doğrulama gerçekleştirin

Numunelerin k–1 serbestlik dereceli bir t dağılımı izlediğini varsayalım (burada, k=10)

t-testi (veya Student t-testi) kullanın

Boş Hipotez: M1 ve M2 aynıdır

Sıfır hipotezini reddedebilirsek, o zaman

M1 ve M2 arasındaki farkın istatistiksel olarak anlamlı olduğu sonucuna varıyoruz

Daha düşük hata oranına sahip modeli seçin

#**Model Selection: ROC Curves**

ROC (Alıcı Çalışma Karakteristikleri) eğrileri: sınıflandırma modellerinin görsel karşılaştırması için

Sinyal algılama teorisinden kaynaklanır

Gerçek pozitif oranı ile yanlış pozitif oranı arasındaki değiş tokuşu gösterir

ROC eğrisinin altındaki alan, modelin doğruluğunun bir ölçüsüdür.

Test gruplarını azalan sırayla sıralayın: pozitif sınıfa ait olma olasılığı en yüksek olan listenin başında görünür

Çapraz çizgiye ne kadar yakınsa (yani alan 0,5'e ne kadar yakınsa), modelin doğruluğu o kadar az olur

Dikey eksen gerçek pozitif oranı temsil eder

Yatay eksen temsilcisi yanlış pozitif oranı

Arsa ayrıca çapraz bir çizgi gösterir

Kusursuz doğruluğa sahip bir modelin alanı 1.0 olacaktır.

#**Issues Affecting Model Selection**

Kesinlik

sınıflandırıcı doğruluğu: sınıf etiketini tahmin etme

Hız

modeli oluşturma süresi (eğitim süresi)

modeli kullanma süresi (sınıflandırma/tahmin süresi)

Sağlamlık: gürültüyü ve eksik değerleri işleme

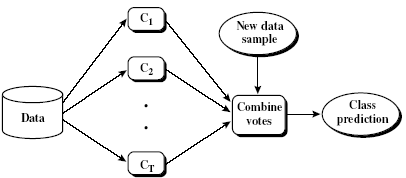
Ölçeklenebilirlik: diskte yerleşik veritabanlarında verimlilik

yorumlanabilirlik

model tarafından sağlanan anlayış ve içgörü

Karar ağacı boyutu veya sınıflandırma kurallarının kompaktlığı gibi kuralların iyiliği gibi diğer ölçütler

**Ensemble Methods: Increasing the Accuracy**



Topluluk yöntemleri

Doğruluğu artırmak için bir model kombinasyonu kullanın

Geliştirilmiş bir M\* modeli oluşturmak amacıyla bir dizi öğrenilmiş modeli, M1, M2, …, Mk'yi birleştirin

Popüler topluluk yöntemleri

Torbalama: bir sınıflandırıcı koleksiyonu üzerinden tahminin ortalamasını alma

Güçlendirme: bir sınıflandırıcı koleksiyonuyla ağırlıklı oylama

Topluluk: bir dizi heterojen sınıflandırıcıyı birleştirmek

#**Bagging: Boostrap Aggregation**

Analoji: Birden fazla doktorun oy çokluğuna dayalı teşhis

Eğitim

D demetlerinden oluşan bir set verildiğinde, her i yinelemesinde, d demetlerinden oluşan bir Di eğitim seti, D'den değiştirilerek örneklenir (yani, önyükleme)

Her eğitim seti Di için bir sınıflandırıcı modeli Mi öğrenilir.

Sınıflandırma: bilinmeyen bir numuneyi sınıflandırın X

Her sınıflandırıcı Mi, sınıf tahminini döndürür

Torbalı sınıflandırıcı M\* oyları sayar ve en çok oyu alan sınıfı X'e atar.

Tahmin: belirli bir test demeti için her bir tahminin ortalama değerini alarak sürekli değerlerin tahminine uygulanabilir.

Kesinlik

Genellikle D'den türetilen tek bir sınıflandırıcıdan önemli ölçüde daha iyi

Gürültü verileri için: çok daha kötü değil, daha sağlam

Tahminde kanıtlanmış geliştirilmiş doğruluk

#**Boosting**

Analoji: Ağırlıklı tanıların bir kombinasyonuna dayalı olarak birkaç doktora danışın—önceki tanı doğruluğuna göre atanan ağırlık

Güçlendirme nasıl çalışır?

Her eğitim demetine ağırlıklar atanır

Bir dizi k sınıflandırıcı yinelemeli olarak öğrenilir

Bir sınıflandırıcı Mi öğrenildikten sonra, sonraki sınıflandırıcı Mi+1'in Mi tarafından yanlış sınıflandırılan eğitim gruplarına daha fazla dikkat etmesini sağlamak için ağırlıklar güncellenir.

Nihai M\*, her bir sınıflandırıcının oylarını birleştirir; burada her bir sınıflandırıcının oyunun ağırlığı, doğruluğunun bir fonksiyonudur.

Yükseltme algoritması, sayısal tahmin için genişletilebilir

Torbalama ile karşılaştırıldığında: Arttırma daha yüksek doğruluğa sahip olma eğilimindedir, ancak aynı zamanda modeli yanlış sınıflandırılmış verilere göre fazla uydurma riski taşır.

#**Adaboost (Freund and Schapire, 1997)**

Bir dizi d sınıfı etiketli demet verildiğinde, (X1, y1), …, (Xd, yd)

Başlangıçta, demetlerin tüm ağırlıkları aynı olarak ayarlanır (1/d)

k turda k sınıflandırıcı üret. i. turda,

D'den demetler, aynı boyutta bir eğitim seti Di oluşturmak için örneklenir (değiştirilerek)

Her demetin seçilme şansı, ağırlığına bağlıdır

Bir sınıflandırma modeli Mi, Di'den türetilmiştir

Hata oranı, test seti olarak Di kullanılarak hesaplanır.

Bir demet yanlış sınıflandırılırsa ağırlığı artar, o.w. azaldı

Hata oranı: err(Xj), Xj demetinin yanlış sınıflandırma hatasıdır. Sınıflandırıcı Mi hata oranı, yanlış sınıflandırılmış demetlerin ağırlıklarının toplamıdır:

#**Random Forest (Breiman 2001)**

Rastgele Orman:

Topluluktaki her sınıflandırıcı, bir karar ağacı sınıflandırıcısıdır ve bölünmeyi belirlemek için her düğümde rastgele bir öznitelik seçimi kullanılarak üretilir.

Sınıflandırma sırasında her ağaç oy verir ve en popüler sınıf döndürülür.

Rastgele Orman oluşturmak için İki Yöntem:

Forest-RI (rastgele giriş seçimi): Her düğümde, düğümdeki bölünme için aday olarak F niteliklerini rastgele seçin. CART metodolojisi, ağaçları maksimum boyuta büyütmek için kullanılır.

Forest-RC (rastgele doğrusal kombinasyonlar): Mevcut niteliklerin doğrusal bir kombinasyonu olan yeni nitelikler (veya özellikler) oluşturur (bireysel sınıflandırıcılar arasındaki korelasyonu azaltır)

Doğruluk açısından Adaboost ile karşılaştırılabilir, ancak hatalara ve aykırı değerlere karşı daha dayanıklı

Her bir bölmede dikkate alınmak üzere seçilen özniteliklerin sayısına duyarsızdır ve torbalama veya güçlendirmeden daha hızlıdır

#**Classification of Class-Imbalanced Data Sets**

Sınıf dengesizliği sorunu: Nadir olumlu örnek, ancak çok sayıda olumsuz örnek, örneğin tıbbi teşhis, dolandırıcılık, petrol sızıntısı, hata vb.

Geleneksel yöntemler, sınıfların dengeli bir dağılımını ve eşit hata maliyetlerini varsayar: sınıf dengesizliği olan veriler için uygun değildir

2 sınıf sınıflandırmasında dengesizlik verileri için tipik yöntemler:

Yüksek hızda örnekleme: pozitif sınıftan verilerin yeniden örneklenmesi

Yetersiz örnekleme: negatif sınıftan demetleri rastgele eleyin

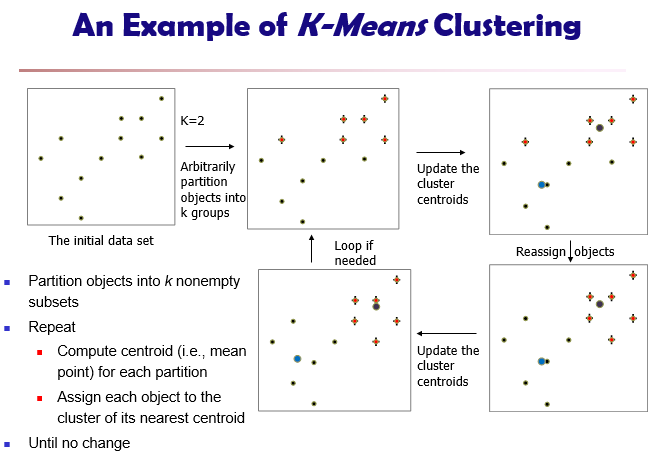
Eşik taşıma: karar eşiğini (t) hareket ettirir, böylece nadir sınıf demetlerinin sınıflandırılması daha kolay olur ve bu nedenle maliyetli yanlış negatif hata olasılığı azalır

Topluluk teknikleri: Yukarıda tanıtılan çoklu sınıflandırıcıları bir araya getirin

Çok sınıflı görevlerde sınıf dengesizliği sorunu için hala zor

**#10.2.1 The *K-Means* Clustering Method**

k verildiğinde, k-means algoritması dört adımda uygulanır:

* Nesneleri boş olmayan k alt kümeye ayırma
* Mevcut bölümlemenin kümelerinin merkezleri olarak tohum noktalarını hesaplayın (merkez merkezi, kümenin merkezidir, yani ortalama noktasıdır)
* Her nesneyi en yakın kaynak noktasına sahip kümeye atayın
* 2. Adıma geri dönün, atama değişmediğinde durum
* 

**#Comments on the *K-Means* Method**

* Güç: Verimli: O(tkn), burada n # nesne, k # küme ve t # yinelemedir. Normalde, k, t << n.
* Karşılaştırma: PAM: O(k(n-k)2 ), CLARA: O(ks2 + k(n-k))
* Yorum: Genellikle yerel bir optimalde sona erer.
* zayıflık
* Yalnızca sürekli n-boyutlu uzaydaki nesnelere uygulanabilir
  + - Kategorik veriler için k-modları yöntemini kullanma
    - Karşılaştırıldığında, k-medoidler geniş bir veri yelpazesine uygulanabilir.
* Küme sayısı olan k'yi önceden belirtmeniz gerekir (en iyi k'yi otomatik olarak belirlemenin yolları vardır (bkz. Hastie ve diğerleri, 2009)
* Gürültülü verilere ve aykırı değerlere duyarlı
* Dışbükey olmayan şekillere sahip kümeleri keşfetmek için uygun değildir

**#Variations of the *K-Means* Method**

Farklı olan k-ortalamalarının varyantlarının çoğu

* İlk k'nin seçilmesi şu anlama gelir:
* Farklılık hesaplamaları
* Küme ortalamalarını hesaplama stratejileri

Kategorik verileri işleme: k-modları

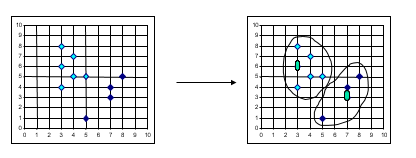
* Küme araçlarının modlarla değiştirilmesi
* Kategorik nesnelerle başa çıkmak için yeni farklılık ölçülerini kullanma
* Küme modlarını güncellemek için frekans tabanlı bir yöntem kullanma
* Kategorik ve sayısal verilerin karışımı: k-prototip yöntemi

**#What Is the Problem of the K-Means Method?**

k-means algoritması aykırı değerlere duyarlıdır!

* Son derece büyük bir değere sahip bir nesne, verilerin dağılımını önemli ölçüde bozabileceğinden

K-Medoids: Bir kümedeki nesnenin ortalama değerini referans noktası olarak almak yerine, bir kümedeki en merkezi konumdaki nesne olan medoidler kullanılabilir.



**#The K-Medoid Clustering Method**

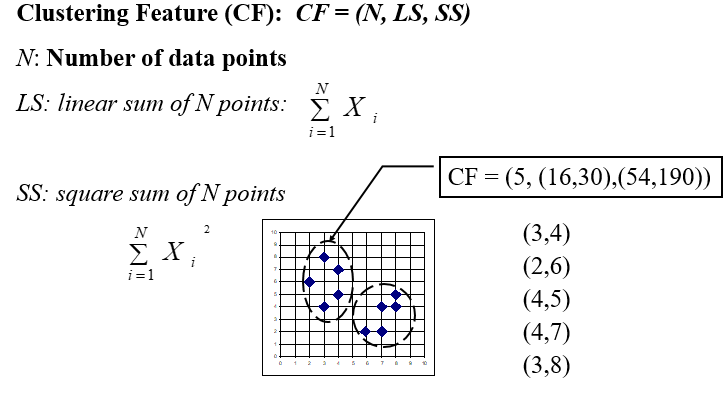
K-Medoids Kümeleme: Kümelerdeki temsili nesneleri (medoidler) bulun

* PAM (Medoidler Etrafında Bölümleme, Kaufmann & Rousseeuw 1987)
  + İlk medoid kümesinden başlar ve ortaya çıkan kümelemenin toplam mesafesini iyileştirirse medoidlerden birini medoid olmayanlardan biriyle yinelemeli olarak değiştirir
  + PAM, küçük veri kümeleri için etkili bir şekilde çalışır, ancak büyük veri kümeleri için iyi ölçeklenmez (hesaplama karmaşıklığından dolayı)

**#BIRCH (Balanced Iterative Reducing and Clustering Using Hierarchies)**

* Zhang, Ramakrishnan & Livny, SIGMOD'96
* Çok fazlı kümeleme için hiyerarşik bir veri yapısı olan bir CF (Kümeleme Özelliği) ağacını artımlı olarak oluşturun
  + Aşama: ilk bellek içi CF ağacını oluşturmak için DB'yi tarayın (verinin doğal kümeleme yapısını korumaya çalışan verilerin çok düzeyli sıkıştırılması)
  + Aşama 2: CF ağacının yaprak düğümlerini kümelemek için rastgele bir kümeleme algoritması kullanın
* Doğrusal olarak ölçeklenir: tek bir taramayla iyi bir kümeleme bulur ve birkaç ek taramayla kaliteyi artırır
* Zayıflık: yalnızca sayısal verileri işler ve veri kaydının sırasına duyarlıdır

**#Clustering Feature Vector in BIRCH**



#CF-Tree in BIRCH

Kümeleme özelliği:

Belirli bir alt küme için istatistiklerin özeti: istatistiksel açıdan alt kümenin 0., 1. ve 2. anları

Bilgi işlem kümesi için önemli ölçümleri kaydeder ve depolamayı verimli bir şekilde kullanır

Bir CF ağacı, hiyerarşik bir kümeleme için kümeleme özelliklerini depolayan, yüksekliği dengeli bir ağaçtır.

Bir ağaçtaki yaprak olmayan bir düğümün torunları veya "çocukları" vardır

Yaprak olmayan düğümler, çocuklarının CF'lerinin toplamlarını saklar.

Bir CF ağacının iki parametresi vardır

Dallanma faktörü: maksimum çocuk sayısı

Eşik: yaprak düğümlerinde saklanan alt kümelerin maksimum çapı

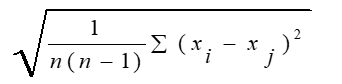
**#The Birch Algorithm**

Girişteki her nokta için

* En yakın yaprak girişini bul
* Noktadan yaprağa giriş ekleyin ve CF'yi güncelleyin
* Giriş çapı > max\_diameter ise, bölünmüş yaprak ve muhtemelen ebeveynler

Algoritma O(n)

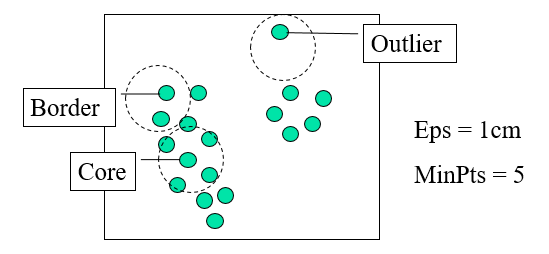
Endişeler

* Veri noktalarının ekleme sırasına duyarlı
* Yaprak düğümlerin boyutunu sabitlediğimiz için kümeler o kadar doğal olmayabilir
* Kümeler, yarıçap ve çap ölçüleri göz önüne alındığında küresel olma eğilimindedir.
* 

#DBSCAN: Density-Based Spatial Clustering of Applications with Noise

Yoğunluğa dayalı bir küme kavramına dayanır: Bir küme, yoğunluğa bağlı noktaların maksimum kümesi olarak tanımlanır

Gürültülü uzamsal veritabanlarında rastgele şekil kümelerini keşfeder



**#DBSCAN: The Algorithm**

Keyfi bir nokta seçin p

p w.r.t.'den yoğunluğa erişilebilen tüm noktaları alın. EPS ve MinPts

p bir çekirdek nokta ise, bir küme oluşur

p bir sınır noktasıysa, p'den yoğunluğa erişilebilen hiçbir nokta yoktur ve DBSCAN veritabanının bir sonraki noktasını ziyaret eder

Tüm noktalar işlenene kadar işleme devam edin

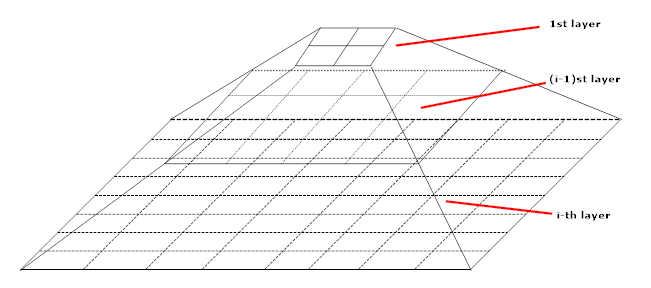
**Grid-Based Clustering Method**

**STING: A Statistical Information Grid Approach**

Wang, Yang ve Muntz (VLDB'97)

Uzamsal alan dikdörtgen hücrelere bölünmüştür

Farklı çözünürlük seviyelerine karşılık gelen birkaç hücre seviyesi vardır.



**#The STING Clustering Method**

Yüksek seviyedeki her hücre, bir sonraki alt seviyede birkaç küçük hücreye bölünür.

Her hücrenin istatistiksel bilgisi önceden hesaplanır ve saklanır ve sorguları yanıtlamak için kullanılır

Daha yüksek seviyeli hücrelerin parametreleri, daha düşük seviyeli hücrenin parametrelerinden kolayca hesaplanabilir

* sayım, ortalama, s, min, maks
* dağılım türü - normal, tek tip, vb.

Uzamsal veri sorgularını yanıtlamak için yukarıdan aşağıya bir yaklaşım kullanın

Önceden seçilmiş bir katmandan (genellikle az sayıda hücreyle) başlayın

Mevcut seviyedeki her hücre için güven aralığını hesaplayın

#STING Algorithm and Its Analysis

Alakasız hücreleri daha fazla değerlendirmeden çıkarın

Geçerli katmanı incelemeyi bitirdiğinizde, bir sonraki alt düzeye geçin

Alt katmana ulaşana kadar bu işlemi tekrarlayın.

Avantajlar:

* Sorgudan bağımsız, paralelleştirmesi kolay, artımlı güncelleme
* O(K), burada K, en düşük seviyedeki ızgara hücrelerinin sayısıdır

Dezavantajları:

* Tüm küme sınırları ya yatay ya da dikeydir ve diyagonal sınır algılanmaz