SAKARYA ÜNİVERSİTESİ

# Veri Madenciliği Uygulamaları

## Hafta 11

**Yrd. Doç.Dr. Nilüfer YURTAY**

# Kümeleme-Bölümlemeli Yöntemler

## 11.1. Giriş

n tane nesnesi olan ve k sayıda küme tanımlanmış bir veritabanı düşünelim. Bu durumda bölümlendirme metodu tüm nesneleri k adet kümeye ayıracaktır. Kümeler, nesneler arasındaki benzersizliklere göre oluşturulur.

Kümeleme, fiziksel ya da soyut nesnelerin benzerliklerine göre gruplanmasıdır. Küme, benzer nesnelerin oluşturduğu bir gruptur. Kümeleme analizi pratikte birçok aktivitede kullanılır. Desen tanımlama, veri analizi, resim işleme, pazar araştırması bunların arasındadır. Kümeleyerek, datalar arasındaki ilginç desenler yakalanabilir.

Pazarlamacıların kendi müşterileri arasındaki farklı grupları karakterize etmesini sağlayabilir. Biyolojide bitki ve hayvan taksonomilerini genlere göre sınıflandırmada kullanılır. Yeryüzü incelemelerinde belli toprak parçalarını tanımlamak için kullanılır. Aynı zamanda web deki dokümanları sınıflamakta kullanılır.

Veri kümeleme çok hızlı bir gelişim içindedir. Uygulama alanları hızlı bir şekilde genişlemektedir. Yıllar geçtikçe analiz edilecek veri miktarı da sürekli arttığı için çok kullanılacak bir yöntemdir.

Kümelemenin sınıflandırmadan farkı sınıflandırmadaki gibi önceden tanımlı sınıf etiketlerinin olmamasıdır. Bu sebeple kümelemede, sınıflandırmadaki gibi örnekleyerek öğrenme yerine gözlemleyerek öğrenme kavramı geçerlidir.

Veri madenciliği alanında kümeleme yapabilmek için bazı gereksinimlerin sağlanmış olması gerekir.

- **Ölçeklendirilebilme:** Kümelendirme algoritması küçük çaplı nesneler üzerinde çalışabilmesine rağmen büyük veriler üzerinde çok performanslı olmayabilir. Bu durumlarda ölçeklendirme algoritmalarına ihtiyaç vardır.
- **Değişik Nesne Tiplerine Göre Çalışabilme:** Günümüzde birçok kümelendirme algoritması sayısal veriler üzerinde çalışması için geliştirilmiştir. Ancak sayısal olmayan ve binary veriler üzerinde de çalışacak algoritmalara ihtiyaç gittikçe artmaktadır.
- **Farklı Tipteki Nesneleri Ayırabilme:** Birçok kümelendirme algoritması nesneler arasında Euclidean ve Manhattan ölçütlerine göre ayrım yapabilmektedir. Bu tür algoritmalar benzer boyuttaki ve benzer yoğunluktaki nesneleri ayırt edebilmektedir. Fakat çok değişik tipte, boyutlarda nesneler olabileceğinden algoritmanın buna uygun olarak çalışması gerekmektedir.

- **En Az Miktarda Alan Bilgisi Gerektirmesi:** Birçok kümeleme algoritması kullanıcı girişlerine ihtiyaç duyar. Kümeleme sonucu da bu parametrelere karşı hassastır ve bunlara göre değişiklik gösterir. Algoritma sonucu parametrelere bu kadar bağımlı olmamalı ve sonuç bu derece hassas olmamalıdır. Bu, parametreyi girecek kullanıcılar için büyük bir sıkıntıdır ve analizin sonucunu kontrol etmeyi zorlaştırır.
- **Çöp Veri Ayıklayabilme:** Gerçek hayatta kullanılan birçok veritabanı; eksik, tanımlanmamış, ayrık veriler içerir. Kümelendirme algoritmaları bu çöp verilerden dolayı kötü sonuçlar verebilir. Bu sebeple, algoritma bu çöp verileri ayıklayabilmelidir.
- **Algoritma, Verilen Parametrelerin Sırasına Duyarsız Olmalıdır:** Bazı algoritmalarda girilen parametrelerin sırası değiştiğinde algoritma sonucu bundan etkilenir. İstenmeyen bu durumun oluşmaması için, algoritmada girilen parametrelerin sırası önemsiz olmalıdır.
- **Yüksek Boyutluluk:** Birçok algoritma 2 ya da 3 boyutlu veriler üzerinde iyi çalışır. İnsan gözü de en çok 3 boyutlu veriyi anlayabilecek yapıdadır. Fakat kümeleme algoritması daha fazla boyutta çalışabilmelidir.
- **Kısıtlama Bazlı Kümeleme:** Günümüz ihtiyaçlarına cevap verebilecek bir algoritma çeşitli kısıtlamalarla çalışabilmelidir. Yani sonuca yansıyacak veriler filtrelenebilmelidir.

Kümeleme algoritmaları eğiticisiz öğrenme metotlarıdır. Örneklere ait sınıf bilgisini kullanmazlar. Temelde verileri en iyi temsil edecek vektörleri bulmaya çalışırlar. Verileri temsil eden vektörler bulunduktan sonra artık tüm veriler bu yeni vektörlerle kodlanabilirler ve farklı bilgi sayısı azalır. Bu nedenle birçok sıkıştırma algoritmasının temelinde kümeleme algoritmaları yer almaktadır.

### 11.2 Klasik Bölümlendirme Metotları: k-means

Kümeleme algoritmalarının en basitidir. Veriyi en iyi ifade edecek K adet vektör bulmaya çalışır. K sayısı kullanıcı tarafından verilir. Nümerik değerler için çalışır[1].

K-Means algoritması, veritabanındaki n tane nesnenin k adet kümeye bölümlenmesini sağlar. Kümeleme sonucu küme içi (intra-cluster) elamanlar arasındaki benzerlikler çok iken, kümeler arası (inter-cluster) elamanları arasındaki benzerlikler çok düşüktür.

---

1 http://209.85.229.132/search?q=cache:SjZGaipi8nsJ:web.firat.edu.tr/daum/docs/62/23%2520Veri%2520Madencili%C4%9Fi%2520-%2520Mehmet%2520Kaya-%C3%B6dendi-6%2520syf-159-164.doc+k+means+k%C3%BCmeleme+%C3%B6rnek&cd=9&hl=tr&ct=clnk&gl=tr

**Kümeleme** sürecinde türüne özgü olarak hata kareleri ölçütü "square-error criterion" toplamı kullanılır.

**E**: veritabanındaki bütün nesnelerin "square error" iki vektör arasındaki uzaklıklarının toplamıdır.
**p:** uzayda bir nesneye verilen noktayı gösterir.
**mi**: $C_i$ kümesinin orta noktasını gösterir.

**Algoritma: K-Means**
**Girdi (Input):**
**k:** küme sayısı
**D:** n tane nesne içeren veritabanı
**Çıktı (output): k** kümesi

***K-means Algoritmasının adımları***

*1. Başlangıçta küme merkezini belirlemek için D veritabanında **k** tane alt küme oluşturulacak şekilde rasgele n tane nesne seçilir.*
*2. Her nesnenin ortalaması hesaplanır. Merkez nokta kümedeki nesnelerin niteliklerinin ortalamasıdır.*
*3. Her nesne en yakın merkez noktanın olduğu kümeye dâhil edilir.*
*4. Nesnelerin kümelemesinde değişiklik olmayana kadar adım 2'ye geri dönülür*

Bu metot ölçeklendirilebilir bir metottur ve çok geniş veritabanları üzerinde de uygulanabilir. Çünkü karmaşıklığı oldukça azdır.

K-Means ; Gerçeklemesi kolaydır ve karmaşıklığı diğer kümeleme yöntemlerine göre azdır.

K-Means algoritması aşağıdaki durumlarda iyi sonuç vermeyebilir:

— Veri grupları farklı boyutlarda ise

— Veri gruplarının yoğunlukları farklı ise

— Veri gruplarının şekli küresel değilse

— Veri içinde aykırılıklar varsa .

## 11.3 Örnek Çalışma[2]

Aşağıdaki 8 nokta için 3 küme elde ediniz.:

A1(2, 10) A2(2, 5) A3(8, 4) A4(5, 8) A5(7, 5) A6(6, 4) A7(1, 2) A8(4, 9).

[2] *faculty.uscupstate.edu/atzacheva/SHIM450/**KMeansExample**.doc*

Ilk küme merkezleri A1(2, 10), A4(5, 8) and A7(1, 2) dir. Iki nokta arasındaki uzaklık değerlerini aşağıdaki formülle hesaplayalım:

*a=(x1, y1)* and *b=(x2, y2)* ; $\rho(a, b) = |x2 - x1| + |y2 - y1|$ .

1.İterasyon

| | | (2, 10) | (5, 8) | (1, 2) | |
|---|---|---|---|---|---|
| | **Nokta** | **1.küme** | **2.küme** | **3.küme** | Küme |
| **A1** | (2, 10) | | | | |
| **A2** | (2, 5) | | | | |
| **A3** | (8, 4) | | | | |
| **A4** | (5, 8) | | | | |
| **A5** | (7, 5) | | | | |
| **A6** | (6, 4) | | | | |
| **A7** | (1, 2) | | | | |
| **A8** | **(4, 9)** | | | | |

nokta merkez1
*x1, y1* *x2, y2*
(2, 10) (2, 10)

$\rho(a, b) = |x2 - x1| + |y2 - y1|$

$\rho(nokta,$ merkez1$) = |x2 - x1| + |y2 - y1|$
$= |2 - 2| + |10 - 10|$
$= 0 + 0$
$= 0$

*x1, y1* *x2, y2*
(2, 10) (5, 8)

$\rho(a, b) = |x2 - x1| + |y2 - y1|$

$\rho(nokta,$ merkez2$) = |x2 - x1| + |y2 - y1|$
$= |5 - 2| + |8 - 10|$
$= 3 + 2$
$= 5$

*x1, y1* *x2, y2*
(2, 10) (1, 2)

$\rho(a, b) = |x2 - x1| + |y2 - y1|$

$\rho(nokta,$ merkez3$) = |x2 - x1| + |y2 - y1|$
$= |1 - 2| + |2 - 10|$
$= 1 + 8$
$= 9$

Elde edilen verileri tabloya yerleştirelim.

1.İterasyon

| | | (2, 10) | (5, 8) | (1, 2) | |
|---|---|---|---|---|---|
| | **Nokta** | **1.küme** | **2.küme** | **3.küme** | Küme |
| **A1** | (2, 10) | 0 | 5 | 9 | **1** |
| **A2** | (2, 5) | | | | |
| **A3** | (8, 4) | | | | |
| **A4** | (5, 8) | | | | |
| **A5** | (7, 5) | | | | |
| **A6** | (6, 4) | | | | |
| **A7** | (1, 2) | | | | |
| **A8** | **(4, 9)** | | | | |

| 1.küme | 2.küme | 3.küme |
|---|---|---|
| (2, 10) | | |

*x1, y1* *x2, y2*
(2, 5) (2, 10)

$\rho(a, b) = |x2 - x1| + |y2 - y1|$

$\rho(nokta, merkez1) = |x2 - x1| + |y2 - y1|$
$= |2 - 2| + |10 - 5|$
$= 0 + 5$
$= 5$

*x1, y1* *x2, y2*
(2, 5) (5, 8)

$\rho(a, b) = |x2 - x1| + |y2 - y1|$

$\rho(nokta, merkez2) = |x2 - x1| + |y2 - y1|$
$= |5 - 2| + |8 - 5|$
$= 3 + 3$
$= 6$

*x1, y1* *x2, y2*
(2, 5) (1, 2)
$\rho(a, b) = |x2 - x1| + |y2 - y1|$

$\rho(nokta, merkez3) = |x2 - x1| + |y2 - y1|$
$= |1 - 2| + |2 - 5|$
$= 1 + 3$
$= 4$

Elde edilen verileri tabloya yerleştirelim.

1.İterasyon

| | | (2, 10) | (5, 8) | (1, 2) | |
|---|---|---|---|---|---|
| | **Nokta** | **1.küme** | **2.küme** | **3.küme** | Küme |
| **A1** | (2, 10) | 0 | 5 | 9 | **1** |
| **A2** | (2, 5) | 5 | 6 | 4 | **3** |
| **A3** | (8, 4) | | | | |
| **A4** | (5, 8) | | | | |
| **A5** | (7, 5) | | | | |
| **A6** | (6, 4) | | | | |
| **A7** | (1, 2) | | | | |
| **A8** | **(4, 9)** | | | | |

| 1.küme | 2.küme | 3.küme |
|---|---|---|
| **(2, 10)** | | **(2, 5)** |

Bu şekilde devam edersek aşağıdaki biçimde tablo tamamlanır.

1.İterasyon

| | | (2, 10) | (5, 8) | (1, 2) | |
|---|---|---|---|---|---|
| | **Nokta** | **1.küme** | **2.küme** | **3.küme** | Küme |
| **A1** | (2, 10) | 0 | 5 | 9 | **1** |
| **A2** | (2, 5) | 5 | 6 | 4 | **3** |
| **A3** | (8, 4) | 12 | 7 | 9 | **2** |
| **A4** | (5, 8) | 5 | 0 | 10 | **2** |
| **A5** | (7, 5) | 10 | 5 | 9 | **2** |
| **A6** | (6, 4) | 10 | 5 | 7 | **2** |
| **A7** | (1, 2) | 9 | 10 | 0 | **3** |
| **A8** | **(4, 9)** | **3** | **2** | **10** | **2** |

| 1.küme | 2.küme | 3.küme |
|---|---|---|
| (2, 10) | (8, 4) | (2, 5) |
| | (5, 8) | (1, 2) |
| | (7, 5) | |
| | (6, 4) | |
| | (4, 9) | |

Yeni küme merkezlerini hesaplayalım:

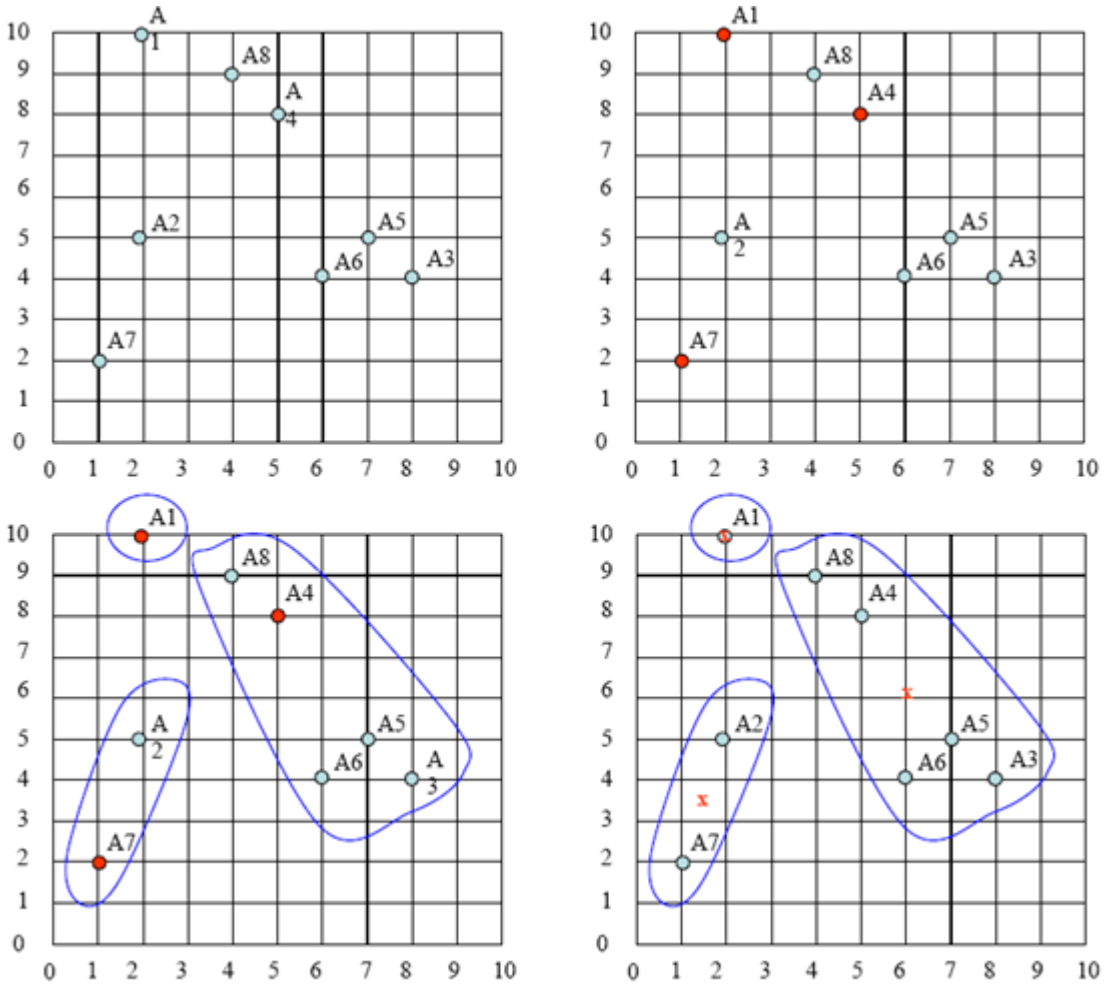
1.küme için A1(2, 10).

2.küme için , ( (8+5+7+6+4)/5, (4+8+5+4+9)/5 ) = (6, 6)

3.küme için , ( (2+1)/2, (5+2)/2 ) = (1.5, 3.5)

Yeni kümeler:
1:{A1}
2:{A3,A4,A5,A6,A8}
3:{A2,A7}

Olarak elde edilmiştir.




**11.4 k-medoids**

Çok yüksek değerdeki nesneler, küme dağılımını olumsuz etkiler. Çünkü k-means tüm değerlere karşı duyarlıdır. k-medoid de, k-means gibi tek tek hesaplamak yerine, her bir küme için kabaca bir temsilci nesne belirlenir (medoid). Kalan her nesneyi bu medoid le karşılaştırır ve benzerliğine göre o nesne kümeye dahil edilir. Bir kümedeki nesneyi alarak, daha yüksek kaliteyi elde edene dek kümeler arasında iteratif olarak yer değiştirme yapılır

*Algoritma:*

1. *k tane nesne seç (medoid)*
2. *tekrarla*
3. *nesneleri onlara en yakın medoidlere at*
4. *medoid olmayan rasgele bir nesne seçilir*
5. *bu nesne bir medoidmiş gibi ele alınıp toplam performans hesaplanır*
6. *eğer daha performanslı sonuç elde ediliyorsa diğeri yerine yeni medoid bu nesne olur (yer değiştirilir)*

   *(örneğin a kümesinden bir nesne seçerek b ve a kümeleriyle karşılaştır ve eğer daha kaliteli bir duruma gelecekse yer değiştir.)*
7. *bir değişiklik olmayana dek tekrarla*

k-medoids, k-means e göre çöp veriden daha az etkilenir.

**11.5 CLARA**

Küçük ölçekli veritabanlarında kullanılan k-medoid yerine büyük veritabanlarında CLARA kullanılır. Temel fikir, tüm veriyi değerlendirmek yerine, tüm veriyi temsil eden ufak bir kesit alınarak analiz yapılmasıdır.

Bu kesit rasgele bir şekilde bulunur. Örneğin 1.000.000 luk bir kayıt dizisinde 100. , 1000. , 1300., 150000. kayıtlar. CLARAN metodunun etkisi ve kalitesi, boyuta ve rasgele seçilen verilerin ne kadar iyi seçildiğine bağlıdır.

CLARA metodu, alınan örnek verilere fazla bağlı olduğu için CLARANS adlı bir metot geliştirilmiştir. CLARANS da örnek bir nesne alınır ve algoritma bir kez geliştirilir, algoritma tekrarlanırken nesne de değiştirilir. CLARANS metodu ile daha kaliteli bir sonuç elde edilir ancak $n^2$ oranında daha maliyetli bir yoldur.