

**T.C
MUĞLA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ**

İSTATİSTİK VE BİLGİSAYAR BİLİMLERİ ANA BİLİM DALI

**BULANIK KÜMELEME ANALİZİ ve BULANIK
MODELLEMeye UYGULAMALARI**




YÜKSEK LİSANS TEZİ

NEVİN GÜLER

MUĞLA 2006

Prof. Dr. Mbariz EMİNOV danıřmanlıęında Nevin GLER tarafından hazırlanan bu alıřma, 03 / 07 / 2006 tarihinde ařaęıdaki jri tarafından İstatistik ve Bilgisayar Bilimleri Anabilim Dalı'nda yksek lisans tezi olarak oybirlięi ile kabul edilmiřtir.

Bařkan : Prof. Dr. Mustafa DİLEK
ye : Prof. DR. Mbariz EMİNOV
ye : Prof. Dr. Novruz ALLAHVERDİ

İmza: 
İmza: 
İmza: 

ÖNSÖZ

Bu çalışmada, Türkiye için Elektrik Tüketim miktarlarının uzun dönemli tahmininin yapılması amaçlanmıştır. Tahmin işlemi için bulanık modelleme yöntemlerinden Anahtarlamalı Bulanık C-Regresyon Yöntemi kullanılmıştır.

Çalışmam sırasında yardımlarını esirgemeyen çok değerli hocam Prof. Dr. Mübariz EMİNOV'a sonsuz teşekkürlerimi sunuyorum. Sevgili arkadaşlarım Araş.Gör. Ebru Yakar, Eşref Deniz, Araş.Gör. Özkan Aslan, Araş.Gör Nida Gökçe, Araş.Gör. Serkan Ballı ve Araş.Gör Aytaç Pekmezci'ye yardımlarından dolayı teşekkür ediyorum. Ayrıca değerli hocalarım Prof. Dr. Mustafa DİLEK ve Yrd.Doç. Dr. Öznur İŞÇİ'ye çalışmalarım sırasında verdikleri destek için çok teşekkür ediyorum.

Son olarak, eşim Ferhat Dincer ve aileme yaşamım boyunca desteklerini ve sevgilerini eksik etmedikleri için teşekkürlerimi sunuyorum.

NEVİN GÜLER

ÖNSÖZ	I
İÇİNDEKİLER	II
ÖZET	V
ABSTRACT	VII
ŞEKİLLER DİZİNİ	IX
TABLolar DİZİNİ	XI
SEMBOLLER ve KISALTMALAR DİZİNİ	XII
SÖZLÜK	XIV
1. GİRİŞ	1
2. KÜMELEME ANALİZİ	3
2.1 Kümeleme Analizine Giriş.....	3
2.2 Kümeleme Uygulamaları	6
2.3 Kümeleme Algoritmalarının Kategorileri	8
2.4 Kümeleme Yaklaşımları	10
2.4.1 Hiyerarşik kümeleme yaklaşımı	10
2.4.1.1 Toplanmış hiyerarşik kümeleme yöntemleri.....	11
2.4.1.2 Bölen kümeleme yöntemleri	15
2.4.2 Bölümlemeli kümeleme yaklaşımı	15
2.4.2.1 Klasik k-ortalamar algoritması.....	17
2.4.2.2 Klasik k-medoid algoritması	19
2.4.3 Yoğunluğa dayalı kümeleme yaklaşımı	22
2.4.4 Izgara tabanlı kümeleme yaklaşımı	23
2.4.5 Bulanık kümeleme yaklaşımı	25
2.5 Kümeleme Algoritmalarının Karşılaştırılması.....	26
2.6 Kümelemenin Doğrulanması	27
2.6.1 Problemin belirlenmesi.....	27
2.6.2 Kümeleme doğrulamanın temel kavramları.....	28
2.6.3 Doğruluk indeksleri	29
2.6.3.1 Harici kriter	29
2.6.3.2 Dahili kriter.....	31
2.6.3.3 Göreli kriter	32
2.6.3.3.1 Klasik kümelemede doğrulama kriteri.....	33
2.6.3.3.2 Bulanık kümelemede doğrulama kriteri.....	37

3. BULANIK MANTIK.....	38
3.1 Bulanık Mantığa Giriş	38
3.2 Bulanık Kümeler	40
3.2.1 Bulanık kümelerin gösterimi	40
3.3 Bulanık küme işlemleri	42
3.3.1 Bulanık kesişim	43
3.3.2 Bulanık birleşim	44
3.3.3 Bulanık tümleyen	44
3.3.4 Kartezyen çarpım	45
3.3.5 Diğer bulanık küme işlemleri	45
3.4 Bulanık Sistemler.....	46
3.4.1 Bulanık sistemlerin yapısı.....	46
3.4.2 Bulanıklaştırma	48
3.4.3 Üyelik fonksiyonlarının belirlenmesi.....	49
3.4.4 Bulanık kurallar	50
3.4.5 Çıkarım mekanizması	51
3.4.6 Durulaştırma	52
4. BULANIK KÜMELEME ALGORİTMALARI.....	55
4.1 Bulanık Kümeleme Yaklaşımları.....	55
4.2 Geleneksel Bulanık Kümeleme Algoritmaları	56
4.2.1 Bulanık c-ortalama algoritması	57
4.2.2 Gustafson-Kessel algoritması	59
4.2.3 Gath-Geva algoritması	62
4.2.4 Hesaplama zamanı	64
4.3 Prototipi Farklı Geometrik Şekle Sahip Kümeleme Algoritmaları.....	65
4.3.1 Bulanık c-regresyon algoritması.....	65
4.3.2 Bulanık c-Hatlar algoritması.....	67
4.3.3 Uyarlamalı bulanık kümeleme algoritması.....	69
4.3.4 Kabuk prototipler	71
4.4 Bulanık Kümelemede Doğrulama	73
4.4.1 Sadece üyelik değerlerini kullanan doğruluk indeksleri	73
4.4.2 Veri seti ve üyelik değerlerini kullanan doğruluk indeksleri.....	74
5. BULANIK MODELLEME.....	78
5.1 Bulanık Modellemeye Giriş.....	78

5.2 Bulanık Modellemenin Ayırt Edici Özellikleri.....	80
5.3 Bulanık Modellemenin Kullanım Alanları.....	81
5.4 Bulanık Model Çeşitleri.....	83
5.4.1 Mamdani model	83
5.4.2 Takagi-Sugeno model.....	85
5.4.3 Sugeno-Yasukawa model	87
5.4.4 Anahtarlamalı bulanık regresyon model.....	89
6. BULANIK KÜMELEMeye DAYANAN BULANIK MODELLERİN GELİŞTİRİLMESİ.....	92
6.1 Geliştirilmiş Takagi-Sugeno Model	92
6.1.1 Modelin yapısının tanımlanması	92
6.1.2 Model parametrelerinin belirlenmesi	95
6.2 Geliştirilmiş Anahtarlamalı Bulanık Regresyon Modeli.....	96
7. GELİŞTİRİLMİŞ BULANIK MODELLER İLE TÜRKİYE’NİN ELEKTRİK TÜKETİMİNİN TAHMİNİ.....	99
7.1 Geliştirilmiş Anahtarlamalı Bulanık Regresyon Modelin Uygulanması... 99	
7.1.1 Adım1: başlangıç değerlerinin girilmesi.....	100
7.1.2 Adım2: model parametrelerinin belirlenmesi.....	103
7.1.3 Adım3: ayrışım matrisinin güncellenmesi.....	104
7.1.4 Adım4: tahmin.....	106
7.2 Geliştirilmiş Takagi-Sugeno Modelleme Algoritmasının Uygulanması... 109	
7.3 Tahmin Modellerinin Değerlendirilmesi.....	112
7.4 Uzun Dönemli Tahmin.....	114
8. TARTIŞMA VE SONUÇLAR.....	116
KAYNAKLAR.....	119
EKLER.....	123
Ek Tablo1.....	123
Ek Tablo2.....	124
Ek1.....	125
Ek2.....	133
Ek3.....	138
Ek4.....	148
ÖZGEÇMİŞ	157

**BULANIK KÜMELEME ANALİZİ ve
BULANIK MODELLEMeye UYGULAMALARI**

(Yüksek Lisans Tezi)

NEVİN GÜLER

MUĞLA ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

2006

ÖZET

Kümeleme analizi, veri setindeki veri noktaları arasındaki benzerlikleri veya benzemezlikleri kullanarak, onları mümkün olduğunca küme içinde homojen ve kümeler arasında ise heterojen gruplara ayırmayı sağlayan yöntemlerden biridir. Bilindiği üzere kümeleme yöntemleri hiyerarşik ve hiyerarşik olmayan kümeleme yaklaşımları olarak ikiye ayrılır. Hiyerarşik kümeleme yaklaşımında, kümeleme veri noktalarının belirli bölümlene düzeylerinde birleştirilmesi ve ayrıştırılması şeklinde yapılır. Hiyerarşik olmayan kümeleme yaklaşımında ise, veri noktaları belli bölümlene kriterine göre önceden belli sayıda kümeye ayrılır.

Bu tezde, hiyerarşik olmayan kümeleme yaklaşımına dayanan ve veri setindeki veri grupları arasında kesin ayrımının söz konusu olmadığı durumlarda başarıyla uygulanan bulanık kümeleme algoritmaları incelenmiştir. Bulanık kümeleme algoritmalarından; bulanık c-ortalamlar ve bulanık c-regresyon kümeleme algoritmalarının bulanık modellemeye uygulamaları ele alınmıştır. Bu bağlamda, Mamdani, Takagi-Sugeno ve Sugeno-Yasukawa gibi çok iyi tanınan bulanık modellerin yapısı ve kurulmaları için gerekli aşamalar açıklanmıştır. Daha sonra Takagi-Sugeno modelinin yeni bir versiyonunun üzerinde durulmuş ve modelin tanımlanması için algoritma adımlarla verilmiştir. Ayrıca, tezde bulanık çoklu regresyon modelinin (anahtarlamalı regresyon modeli) iyileştirilmesi konusu da ele alınmış ve bu amaçla yeni modelleme algoritması sunulmuştur.

Son olarak, söz konusu iki bulanık modelin iyileştirilmiş versiyonu Türkiye'nin elektrik tüketiminin tahminine uygulanmıştır. Bu modellerle bulunan tahmin sonuçları önceden kullanılan diğer tahmin metotlarının sonuçları ile karşılaştırılmıştır. Elde edilen sonuçların önceki tahminlerden daha iyi olduğu anlaşılmıştır.

Anahtar Kelimeler: kümeleme analizi, bulanık küme, üyelik fonksiyonu, doğruluk indeksi, bölümlene matrisi, regresyon model, bulanık kural, tahmin.

Sayfa Adedi: 157

Tez Yürütücüsü: Prof. Dr. Mübariz EMİNOV

**FUZZY CLUSTERING ANALYSIS and
APPLICATIONS to FUZZY MODELLING**

(Ms. Thesis)

NEVİN GÜLER

MUĞLA UNIVERSITY

INSTITUTE of SCIENCE and TECHNOLOGY

2006

ABSTRACT

Clustering Analysis is one of the methods which enable to separate data points of data set into groups which should be homogeneous within clusters and heterogeneous between them as possible through the use similarities or dissimilarities between data points. As known, clustering methods are conducted by hierarchical and non-hierarchical approaches. In the hierarchical approach, clustering is done by merging and disjointing data points at certain levels of partition. In the non-hierarchical clustering approach, however, data points are divided into the predetermined number of clusters according to predefined partition criterion.

In the present thesis, fuzzy clustering algorithms based on non-hierarchical clustering approach were studied which are utilized successfully when there is no sharp separation between data groups. Applications of the fuzzy clustering algorithms such as fuzzy c-means and fuzzy c-regression to fuzzy modeling have been comprehensively considered. In this context, the structure and the necessary model building stages of widely known fuzzy models such as Mamdani, Takagi-Sugeno and Sugeno-Yasukawa were explained. Then, the development of new version of Takagi-Sugeno model and algorithm for this model identification are given. Furthermore, in this thesis, the improvement of fuzzy multiple regression model (switching regression model) is handled and for this aim, the new modeling algorithm is presented.

Finally, the improved version of the two considered fuzzy models have been applied to forecasting of electric energy for Turkey. The results obtained by these models are compared with results got by methods applied previously. It is seen that the forecasting performance of the models are better than these early methods.

Keywords: Clustering analysis, fuzzy set, membership function, validity index, partition matrix, regression model, fuzzy rule, forecasting.

Total Page: 157

Thesis Director: Prof. Dr. Mübariz EMİNOV

ŞEKİLLER DİZİNİ

<u>Şekil No</u>	<u>Sayfa No</u>
Şekil2.1 :Kümeleme Sürecinin Adımları.....	5
Şekil 2.2 :Bölümlemeli Kümelemenin Akış Diyagramı.....	15
Şekil 2.3 :k-ortalamlar Yöntemi.....	18
Şekil 2.4 :k-medoids Yöntemi.....	20
Şekil 2.5 :(a) 3 küme (b) 4 küme K-ortalamlar'dan elde edilen sonuçlar.....	28
Şekil 2.6 :(a) İki-yönlü indeks (b)Sağ-Yönlü indeks (c) sol-yönlü indeks.....	30
Şekil 3.1 :Yaygın olarak kullanılan üyelik fonksiyonlar.....	41
Şekil 3.2 :A ve B bulanık kümelerine ilişkin üyelik fonksiyonları.....	43
Şekil 3.3 :A ve B bulanık kümelerinin kesişimi.....	43
Şekil 3.4 :A ve B bulanık kümelerine ilişkin birleşim işlemi.....	44
Şekil 3.5 :A bulanık kümesi için tümleyen işlemi.....	45
Şekil 3.6 :Genel bulanık sistem.....	47
Şekil 4.1 :BCO Algoritması Sonucunda Elde Edilen Kümeler.....	58
Şekil 4.2 :(a) BCO Algoritması (b)GK Algoritması.....	60
Şekil 4.3 :GG Algoritmasının Uygulanması Sonucunda Elde Edilen Kümeler.....	64
Şekil 4.4 :BCO ve BCH kümeleme algoritmaları.....	68
Şekil 4.5 :BCH Kümeleme Algoritması.....	68
Şekil 4.6 :UBKA Uygulanması Sonucunda Elde Edilen Kümeler.....	70
Şekil 4.7 :Gustafson-Kessel Analizi.....	70
Şekil 4.8 :BCK Analizi.....	72
Şekil 4.9 :BCKK Analizi.....	73
Şekil 5.1 :Klasik, aralık ve bulanık argümanlar için, klasik aralık ve bulanık fonksiyonların değerlendirilmesi.....	82
Şekil 5.2 :Mamdani tipi bulanık model.....	84
Şekil 5.3 :T-S Bulanık Model.....	86
Şekil 5.4 :Sugeno-Yasukawa Bulanık Model.....	88
Şekil 5.5 :Anahtarlama.....	90

Şekil 7.1 :Farklı küme sayıları için V_u ve V_o Grafiği.....	101
Şekil 7.2 :Kümelere göre tahmin değerleri.....	106
Şekil 7.3 :BCRM yönteminin eğitim seti için performansı.....	108
Şekil 7.4 :BCRM yöntemi için modellemenin yüzde hatası.....	108
Şekil 7.5 :T-S yönteminin eğitim seti için performansı.....	112
Şekil 7.6 :T-S modelleme yöntemi için yüzde hatalar.....	112

TABLolar/ÇİZELGELER DİZİNİ

<u>Tablo No</u>	<u>Sayfa No</u>
Tablo5.1 :Farklı Modelleme Paradigmaları.....	79
Tablo5.2 :Sistemlerdeki klasik ve bulanık bilgi.....	82
Tablo7.1 :Farklı Küme Sayıları için Doğruluk İndeksi V_{sv}	100
Tablo7.2.:BCO Algoritması Sonucunda Elde Edilen Üyelik Değerleri.....	102
Tablo7.3 :BCO Algoritması Uygulandıktan Sonra Elde Edilen Kümeler.....	103
Tablo7.4 :Küme Merkezleri.....	103
Tablo7.5 :BCRM Algoritmasında sonucunda elde edilen üyelik değerleri.....	105
Tablo7.6:BCRM algoritması sonucunda elde edilen kümeler.....	106
Tablo7.7:1970-1990 Yıllarına İlişkin Tahmin ve MYH Değerleri.....	107
Tablo7.8:T-S Modeli Standart Sapma ve Küme Merkezi Değerleri.....	109
Tablo7.9:T-S modelleme yöntem için elde edilen üyelik değerleri.....	110
Tablo7.10:TS Bulanık Modelleme yöntemine göre elde edilen tahmin, gerçek ve MYH değerleri.....	111
Tablo7.11:1991-1998 Yıllarına İlişkin Tahmin Değerleri.....	113
Tablo7.12:1991-1998 Yıllarına İlişkin MYH(%) ve OMYH.....	113
Tablo7.13:1999-2002 Yıllarına İlişkin Tahmin Değerleri.....	114
Tablo7.14 :1999-2002 Yıllarına İlişkin MYH(%) ve OMYH.....	114
Tablo7.15:2003-2010 Uzun Dönemli Tahmin Sonuçları.....	115

SEMBOLLER VE KISALTMALAR DİZİNİ

\tilde{y}_k	:Bulanık Çıktı
β	:Parametre Vektörü
U_{ij}	:i. verinin j. kümeye üyelik derecesi
d_{ij}	:İki vektör uzaklık
ε	:İşlem Bitirme Kriteri
Σ_j	:j. kümenin kovaryans matrisi
λ	:Özdeğer Vektörü
T_{ih}	:Toplam yer değiştirme katkısı
$diam(C)$:Kümelerin dağılım ölçüsü
A	:Simetrik ve pozitif tanımlı matris
ABCRM	:Anahtarlamalı Bulanık C-Regresyon Model
$A^i(x)$:x. verisinin i. kümeye Gaussian Üyelik Derecesi
BCE	:Bulanık C- Elliptotype'lar
BCH	:Bulanık C-Hatlar
BCK	:Bulanık C-Kabuklar
BCKK	:Bulanık C-Küresel Kabuklar
BCO	:Bulanık C-Ortalamalar
BCR	:Bulanık C-Regresyon
BCRM	:Bulanık C-Regresyon Model
BEK	:Bölünme Entropi Koefsiyanı
BK	:Bulanık Kümeleme
c	:Küme Sayısı
d_{min}	:Kümeler Arasındaki Uzaklığının minimumu
E(C)	:Uzaklık Ölçüsü
EKKY	:En Küçük Kareler Yöntemi
F	:Kovaryans Matrisi
GAKT	:Gruplar arası kareler toplamı
GGA	:Gath-Geva Algoritması
GKA	:Gustafson-Kessel Algoritması
GKT	:Grup içi kareler toplamı
ITK	:Izgara Tabanlı Kümeleme

θ_i	:i kümenin koefsiyanları
J	:Amaç Fonksiyonu
KA	:Kümeleme Analizi
KMA	:K-Medoid'ler Algoritması
KOA	:K-Ortalamalar Algoritması
KOKSS	:Kok-Ortalama-Kare-Standart-Sapması
KP	:Kabuk Prototip
KU	:İki küme arasındaki uzaklık
m	:Bulanıklık indeksi
MBKY	:McOuity Bağlantılı Kümeleme Yöntemi
MD _i	:Küme İçi Uzaklık Ortalaması
MerBKY	:Merkezsiz Bağlantılı Kümeleme Yöntemi
MYH	:Mutlak Yüzde Hata
OBKY	:Ortanca Bağlantılı Kümeleme Yöntemi
OMYH	:Ortalama Mutlak Yüzde Hata
OrtBKY	:Ortalama Bağlantılı Kümeleme Yöntemi
RK	:R kare
RT	:Regresyon Tekniği
SCO	:Sert C-Ortalamalar
TAI	:Tusaş Havacılık ve Uzay Sanayi
TBKY	:Tam Bağlantılı Kümeleme Yöntemi
TekBKY	:Tek Bağlantılı Kümeleme Yöntemi
TK	:Toplam kareler toplamı
T-S	:Takagi-Sugeno
UBK	:Uyarlamalı Bulanık Kümeleme
v_i	:i kümenin prototipi
V_o	:Küme Arası Hata
V_{sv}	:Doğrulama İndeksi
V_u	:Küme İçi Hata
WBKY	:Ward Bağlantılı Kümeleme Yöntemi
XB	:Xie-Beni
YDK	:Yoğunluğa Dayanan Kümeleme
YKRR	:Yarı-Kısmi R Kare
YSA	:Yapay Sinir Ağları

SÖZLÜK

Adaptive	:Uyarlanabilir
Additive	:Katkı
Agglomerative	:Toplanmış
Algorithm	:Algoritma
Approximation	:Yakınsayıcı
Average Partition Density	:Ortalama Bölünme Yoğunluğu
Boolean	:Klasik
Cartesian Product	:Kartezyen Çarpım
Center	:Merkez
Centroid	:Küme Merkezi
Clustering Analysis	:Kümeleme Analizi
Clustering Validation	:Kümeleme Doğrulama
Complement	:Tümleyen
Consequent	:Sonuç
Crisp	:Klasik
Data-Fitting	:Veri Uydurma
Defuzzification	:Durulaştırma
Density	:Yoğunluk
Density-Based Clustering	:Yoğunluğa Dayalı Kümeleme
Divise	:Bölen
Equality	:Eşitlik
Expansion	:Genişleme
External Criterion	:Harici Kriter
Forecasting	:İleriye Dönük Tahmin
Furhest Neighbor Complete Linkage:	En Uzak Komşuluk Tam Bağlantılı
Fuzzification	:Bulanıklaştırma
Fuzzy Logic	:Bulanık Mantık
Fuzzy Modeling	:Bulanık Modelleme
Fuzzy Rule	:Bulanık Kural
Fuzzy Scatter Matrix	:Bulanık Serpilme Matrisi
Fuzzy Set Operation	:Bulanık Küme İşlemi

Fuzzy Sets	:Bulanık Kümeler
Fuzzy Systems	:Bulanık Sistemler
Fuzzy Clustering:	:Bulanık Kümeleme
Grid	:Izgara
Grid-Based Clustering	:Izgara Tabanlı Kümeleme
Inference Mechanism	:Çıkarım Mekanizması
Input	:Girdi
Internal Criterion	:Dahili Kriter
Intersection	:Kesişim
Interval	:Aralık
Knee	:Diz
Line-shaped	:Çizgi-biçimli
Linguistic Variable	:Dilsel Değişken
Linkage Metric	:Bağlantılı-Ölçülü
Maximum Probability Estimator:	Maksimum Olabilirlik Tahmin Edicisi
Membership Function	:Üyelik Fonksiyonu
Minimum Spanning Tree	:Minimum Karışlama Ağacı
Multi-resolution	:Çoklu-ayırıştırma
Nearest Neighbor-Single Linkage:	En Yakın Komşuluk Tek Bağlantılı
Objective Function	:Amaç Fonksiyonu
Outlier	:Dışarıda kalanlar, Aykırı değerler
Output	:Çıktı
Partition Entropy Coefficient:	Bölünme Entropi Katsayısı
Partition	:Bölünme
Posteriori	:Sonsal
Premise	:Öncül
Priori	:Önsel
Problem Specification	:Problemin Belirlenmesi
Relative Criterion	:Görelî Kriter
Separation	:Ayırma
Shell	:Kabuk
Singleton	:Bulanık Tekillik

Statistical Information Grid	:İstatistiksel Bilgi Izgarası
Straight Line	:Düz Hat
Switching	:Anahtarlama
Union	:Birleşim
Validity Index	:Doğruluk İndeksi

1.GİRİŞ

Kümeleme analizi, son zamanlarda iş ve bilim alanında sıkça kullanılmaya başlanan çok değişkenli veri analiz yöntemlerinden biridir. Bu yöntemin temel amacı, veri setindeki veri noktalarını benzerliklerine göre alt gruplara ayırmaktır. Başka bir ifade ile veri noktaları arasındaki benzerlikler dikkate alınarak benzer veri noktalarının aynı grupta veya kümede toplanmasını sağlamaktır.

Kümeleme analizi yöntemleri temel olarak hiyerarşik ve hiyerarşik olmayan olmak üzere iki yaklaşıma ayrılmaktadır. Hiyerarşik kümeleme yaklaşımında, veri noktaları optimal kümeleme yapısı elde edilene kadar belirli düzeylerde birleştirilir ya da ayrıştırılır. Bu algoritmalar için küme sayısının önceden bilinmesine gerek yoktur. Hiyerarşik olmayan kümeleme yaklaşımları ise gerek teorik dayanaklarının daha güçlü olması, gerekse küme sayısı konusunda ön bilgi olması ya da araştırmacının anlamlı olacak küme sayısına karar verebilmesi açısından tercih edilmektedir.

Bu çalışmada hiyerarşik olmayan kümeleme yaklaşımlarına dayanan bulanık kümeleme algoritmaları ve bu algoritmalarından Bulanık C-Ortalamlar ve Bulanık C-Regresyon algoritmalarının bulanık modellemeye uygulamaları ele alınmıştır. Ayrıca, bulanık kümeleme analizinin esnek ve yorum açısından kullanışlı olması, konu ile ilgili uzman bilgisini dikkate alması, istatistiksel yöntemlerden farklı olarak herhangi bir varsayıma dayanmaması gibi avantajları esas alınarak bu analize dayanan farklı kümeleme algoritmaları da incelenmiştir.

Bulanık modelleme ise, bulanık modelin yapısıyla ve ilgili model parametreleriyle sistem tanımlamasının yeni bir koludur ve örnek bir veri seti ile tanımlanan, yapısı bilinmeyen bir sistemin davranışını tahmin eder ve açıklar. Klasik matematiğe dayanan sistem modellemesi, eksik tanımlanmış ve belirsiz sistemler için pek uygun değildir. Bunun aksine, bulanık mantığa dayanan modelleme sistemi, hassas nicel analizler kullanmadan insan bilgisinin ve yaklaşım süreçlerinin nitel taraflarını modelleyebilir. Son yıllarda yeni konular arasında ilk sırayı tutan bulanık küme, bulanık mantık ve bulanık sistemler hemen her mühendislik dalında uygulanır hale gelmiştir. Şu ana kadar bulanık mantığa dayanan modelleme işlemi için Mamdani, Bulanık C-Regresyon, Takagi-Sugeno, Sugeno-Yasukawa bulanık model

vs. gibi deęişik yaklaşımlar geliştirilmiştir. Bu bulanık modelleme yöntemleri, sistem hakkında eksik bilgilerin veya belirsizliğin olması durumunda bile sistem davranışını tahmin edebilme yeteneğine sahiptirler. Fakat bu bulanık modellerin oluşturulabilmesi için kural sayısının (Küme sayısı), başlangıç ayrışım matrisinin ve modelde yer alacak deęişkenlerin ne olacağına önceden karar verilmesi gerekmektedir. Şu ana kadar yapılan çalışmalarda, bu tür başlangıç koşullarının seçimi, en uygun model yapısı ve model parametreleri elde edilene kadar sürecin defalarca tekrarlanmasına dayanmaktaydı. Bu nedenle, bu çalışmada, başlangıç koşullarının seçimi için doğruluk indeksleri ve bulanık kümeleme algoritmaları kullanılarak bu tür problemlerin ortadan kaldırılması amaçlanmıştır.

Bu doğrultuda, adı geçen bulanık modellerden Takagi-Sugeno ve Anahtarlamalı Bulanık Regresyon modellerinin, Bulanık C-Ortalamalar ve Bulanık C-Regresyon kümeleme algoritmaları kullanılarak geliştirilmesi üzerinde durulmuştur.

Ayrıca bu çalışmada, bahsedilen bu iki geliştirilmiş bulanık model Türkiye'nin elektrik tüketim miktarı verilerine uygulanmış ve elde edilen sonuçlar Regresyon Teknięi, Box-Jenkins ve Yapay Sinir Ağları gibi dięer modelleme tekniklerinden elde edilen sonuçlarla karşılaştırılmıştır.

2. KÜMELEME ANALİZİ

2.1 Kümeleme Analizine Giriş

Kümeleme Analizi (KA), bir araştırmada incelenen birimleri aralarındaki benzerliklerine göre belirli gruplar içinde toplayarak sınıflandırma yapmayı, birimlerin ortak özelliklerini ortaya koymayı ve bu sınıflar ile ilgili genel tanımlamalar yapmayı sağlayan bir yöntemdir. Analiz sonucu elde edilen kümeler yüksek düzeyde küme içi homojenlik ve yüksek düzeyde kümeler arası heterojenlik gösterirler (Sharma, 1996).

Veri: Kümeleme teknikleri, kategorik, sayısal veya her ikisinin karışımı olan verilere uygulanabilir. Veri, genellikle fiziksel bazı süreçlerin gözlemleridir. Her bir gözlem n boyutlu satır vektörü olarak gruplanan n tane ölçüm içerir. $X_k = [x_{k1}, x_{k2}, \dots, x_{kn}]^T, X_k \in R^n$ N tane gözlemin kümesi $X = \{ X_k | k = 1, 2, \dots, N \}$ şeklinde gösterilir. Desen tanıma terminolojisinde, X 'in satırları desen veya birey olarak, sütunları ise özellik olarak adlandırılır ve X matrisine desen matrisi denir.

Kümeler: Kümelemenin amacına göre kümelerin farklı tanımları yapılabilir. Genel olarak kabul edilen bir tanım şu şekildedir; Küme diğer kümelerdeki bireylere göre birbirlerine daha benzer olan bireylerin oluşturduğu gruptur. Buradaki benzerlik matematiksel benzerlik olarak anlaşılabilir. Kümeler farklı boyutlarda, geometrik şekillerde ve yoğunlukta olabilir (Balasko vd., URL5).

Kümeleme Analizi de X veri matrisinde yer alan ve doğal gruplamaları kesin olarak bilinmeyen bireyleri, değişkenleri ya da birey ve değişkenleri birbirleri ile benzer olan alt kümelere ayırmaya yardımcı olan yöntemler topluluğudur. KA; birimleri p değişkene göre hesaplanan ve benzerlik ölçüsü olarak kullanılan bazı ölçüler kullanarak homojen gruplara bölmek amacıyla kullanılır. Bu amaçlar dört grupta toplanabilir.

- n sayıda birimi, nesneyi, oluşumu, p değişkene göre saptanan özelliklerine göre olabildiğince kendi içinde türdeş (homojen) ve kendi aralarında farklı (heterojen) alt gruplara (küme) ayırmak.

- p sayıda değişkeni, n sayıda birimde saptanan değerlere göre ortak özellikleri açıkladığı varsayılan alt kümelere ayırmak ve ortak faktör yapıları ortaya koymak.

- Hem birimleri hem deęişkenleri birlikte ele alarak, ortak n birimi p deęişkene göre ortak özellikli alt kümelere ayırmak.
- Birimleri, p deęişkene göre saptanan deęerler için, izledikleri biyolojik ve tipolojik sınıflamayı ortaya koymak (taksonomik sınıflandırma yapmak).

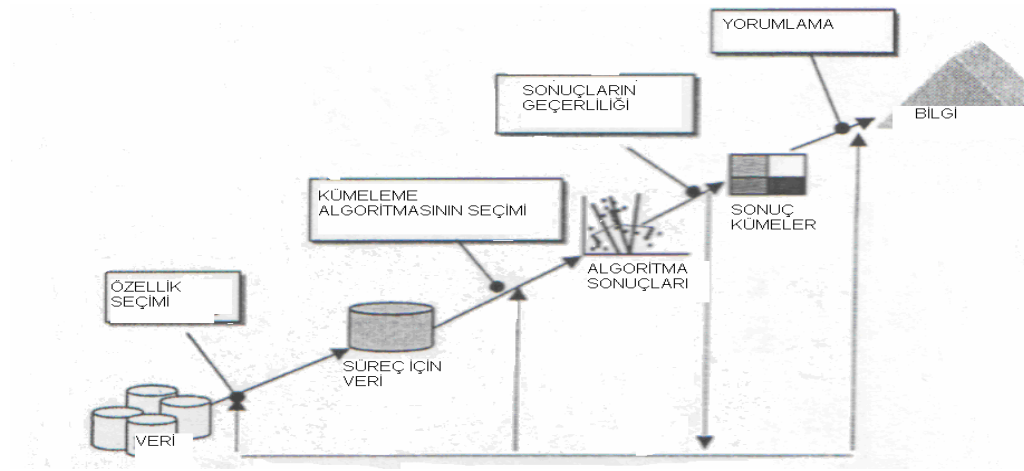
KA, kümelerin sayısına veya küme yapılarına ilişkin herhangi bir varsayımda bulunmaz. Dięer çok deęişkenli istatistiksel analiz yöntemlerinde önemli bir yer tutan normallik varsayımı, bu analizde prensipte kalmakta ve uzaklık deęerlerinin normallięi yeterli görülmektedir.

Kümeleme analizinin uygulama aşamaları aşıęıdaki gibi verilebilir.

- Birim ya da deęişkenlerin doęal gruplamaları hakkında kesin bilgilerin bulunmadıęı popülâsyonlardan alınan n sayıda birimin p sayıda deęişkenine ilişkin gözlemlerin elde edilmesi (Veri matrisinin belirlenmesi).
- Birimlerin/deęişkenlerin birbirleri ile olan benzerliklerini ya da farklılıklarını gösteren uygun bir benzerlik ölçüsü ile birimlerin/deęişkenlerin birbirlerine uzaklıklarının hesaplanması (Benzerlik ya da farklılık matrisinin belirlenmesi).
- Uygun kümeleme yöntemi yardımı ile benzerlik/farklılık matrislerine göre birimlerin/deęişkenlerin uygun sayıda kümelere ayrılması.
- Elde edilen kümelerin yorumlanması ve bu kümeleme yapısına dayalı olarak kurulan hipotezlerin doęrulanması için gerekli analitik yöntemlerin uygulanması (Şahin, 2002).

Kümeleme analizi, iki gözlemin benzerlikleri veya farklılıkları temel alınarak yapılır. Uzaklık ölçüleri ya da benzerlik ölçüleri veri matrisinde yer alan deęişkenlerin ölçü birimlerine göre de farklılık göstermektedir. Eęer deęişkenler oransal ya da aralıklı ölçekle elde edilmiş deęerler ise uzaklık ya da ilişki türü ölçülerden yararlanılır. Ölçümler sayısal deęerler olarak yapılmış ise tercih edilen ölçüler ki-kare uzaklık ölçüsü ya da Phi kare uzaklık ölçüsüdür. Eęer ikili (binary) gözlemlere göre ölçümler yapılmış ise birimler arasındaki benzerlikleri belirlemede Öklid, kare Öklid, boyut farklılıęı (size difference), desen farklılıęı (pattern difference), Lance ve Williams farklılıęı, şekil farklılıęı (shape difference) gibi benzerlik ya da farklılık ölçülerinden yararlanılmaktadır (Özdamar, 2002).

Yaygın olarak kullanılan kümeleme yöntemleri birimler arasındaki uzaklıklara dayanan benzerlik ya da benzemezlik matrisine göre işlem yaptıklarından, farklı kümeleme yöntemleri farklı uzaklık ölçülerine göre farklı sonuçlar verebilmektedir. Ayırmaya dayanan kümeleme yöntemlerinden bazıları her veri setinin her bir birimini bir ve yalnızca bir kümeye ayırır. Böylelikle hiyerarşik ya da bazı hiyerarşik olmayan kümeleme yöntemleri her bir birim için kesin karar alırlar ve bir kümeye atarlar. Sonuçları itibariyle yaklaşık aynı sonuçları veren kümeleme algoritmalarında bazı birimlerin farklı kümelerde yer aldığı gözlemlenmektedir (Hamarat, 1998).



Şekil 2.1 Kümeleme Sürecinin Adımları

Kümeleme süreci, kümelemede kullanılan özel kritere bağlı olarak, veri setinin farklı bölünmeleri ile sonuçlanabilir. Bu yüzden, kümelemeden önce bir takım işlemler yapmak gerekir. Kümeleme sürecini geliştirmenin Şekil 2.1'den görüldüğü gibi basit adımları aşağıdaki gibidir.

Özellik Seçimi: Amaç, ilgilenilen konuda mümkün olduğu kadar çok bilgiyi kodlayabilen, kümeleme ile ilgili özellikleri doğru dürtüst bir şekilde seçmektir. Bu yüzden, verilerin kümeleme adımlarından önce işlenmesi gerekli olabilir.

Kümeleme Algoritması: Bu adım, veri seti için iyi bir kümeleme tasarımının tanımından ortaya çıkan algoritmanın seçimiyle ilgilidir. Yakınlık ölçüsü ve

kümeleme kriteri çoğunlukla, veri setinin yapısına uygun kümeleme tasarımını tanımlamak için oldukça hızlı ve verimli çalışan kümeleme algoritmasını niteler (karakterize eder).

i) Yakınlık Ölçüsü: İki veri noktasının birbirlerine ne kadar benzer olup olmadıklarını ölçer. Çoğu durumda veri noktalarının tüm özellikleri yakınlık ölçüsünün hesaplanmasında eşit katkıda bulunur. Veri noktalarının hiç bir özelliği diğerlerine göre baskın değildir.

ii) Kümeleme Kriteri: Bu adımda, sabit bir fonksiyon ile ya da diğer bazı kural çeşitleri vasıtasıyla ifade edilebilen kümeleme kriterini tanımlamak gerekir. Veri seti içerisinde meydana gelmesi beklenen tüm küme çeşitlerinin göz önüne alınması gerekir. Böylece, veri setine uygun bölünmeyi sağlayan en iyi kümeleme kriterini tanımlayabiliriz.

Sonuçların Geçerliliği: Kümeleme algoritmasının sonuçlarının doğru olup olmadığı uygun kriter ve tekniklerle test edilebilir. Mademki, kümeleme algoritmaları önceliği bilinmeyen kümeleri tanımlar, o zaman kümeleme metotlarına bakılmaksızın, verinin sonuç bölünmesi çoğu uygulamada bazı değerlendirmeler gerektirir.

Sonuçların Yorumu: Birçok durumda, uygulama alanındaki uzman kişiler doğru karara varmak için diğer deneysel kanıtları da göz önüne alarak küme sonuçlarını değerlendirmek zorundadır.

2.2 Kümeleme Uygulamaları

Kümeleme analizi iş ve bilim dallarındaki birçok uygulama için ana araçtır. Bu vesile ile aşağıda, kümelemeyi kullanan basit uygulamalara değinilmiştir.

- **Veri Azaltma:** Kümeleme analizi, bilgi içeren verinin sıkıştırılmasına katkıda bulunabilir. Birkaç farklı durumda, elde edilen veri çok büyük olabilir ve bu verinin işlenmesi çok dikkat gerektirebilir. Kümeleme veri setini ilgilenilen sayıda kümeye bölmek için kullanılabilir. Daha sonra, veri seti, tek bir küme gibi işlenmek yerine, sürecimizin içinde tanımlanan kümelerin örneği gibi benimsenir ve böylece veri sıkıştırılması başarılı olur.
- **Hipotez Oluşturma:** Kümeleme analizi, burada veriyi ilgilendiren bazı hipotezler oluşturmak için kullanılır. Örneğin, bir alışveriş merkezini ziyaret eden

müşterilerin yaşlarıyla ile onların satın alma saatlerini bir veri tabanında tuttuğumuzu düşünelim. Bu verileri kullanılarak şu şekilde bir hipotez kurulabilir.

Genç insanlar akşam saatlerinde alışveriş yapmaktadır

Yaşlı insanlar sabah saatlerinde alışveriş yapmaktadır.

- **Hipotez Testi:** Bu durumda, kümeleme analizi açıkça belirtilen bir hipotezin doğruluğunu sağlamak için kullanılabilir. Örneğin, aşağıdaki gibi bir hipotez olduğu düşünülün “*Genç insanlar akşam saatlerinde alışveriş yapmaktadır*”. Bunun doğru olup olmadığını sağlamanın bir yolu örnek veri setine kümeleme analizini uygulamaktır. Her verinin müşterinin yaşını, işini vs. temsil ettiği düşünülürse ve kümeleme analizi uygulanırsa, genç insanların akşam satın almalarına karşılık gelen küme oluşturulmuş olacaktır. Böylece hipotez kümeleme analizi tarafından desteklenir.

- **Gruplara Dayanan Öngörü:** Kümeleme analizi veri setine uygulanır ve kümelere giren bireylerin özelliklerini karakterize eden kümelerinin oluşumunu sağlar. Daha sonra bilinmeyen bireyler bu kümelere olan uzaklıklarına veya benzerliklerine göre herhangi bir kümeye dâhil edilir. Buradan ilgilenilen verilere ilişkin kullanışlı bilgiler çıkarılabilir. Örneğin aynı hastalığı geçiren hastalarla ilgilenen bir veri seti olduğu varsayalım. Sonuçta, hastaların verilen ilaçlara gösterdiği farklı tepki sayısı kadar küme oluşacaktır. Kümeleme yapıldıktan sonra birkaç yeni hasta incelenecek ve bu hastalar verilen ilaca gösterdiği tepkiye göre, önceden belirlenen bu kümelerden herhangi birine girecektir.

Kümelemenin genel olarak kullanılan bazı uygulamaları da aşağıdaki gibidir:

İş: İş sektöründe, kümeleme, müşteri veritabanlarına göre önemli pazar gruplarını belirlemeye yardımcı olabilir.

Biyoloji: Biyolojide, benzer fonksiyonel özellikleri taşıyan genleri kategorize etmede ve popülasyonun doğasını kavramada kullanılabilir.

Uzayla İlgili Veri Analizi: Uydu görüntülerinden, tıbbi aletlerden, Coğrafi Bilgi Sistemleri’nden (CBS), görüntü veri tabanı araştırmalarından vs. elde edilebilen uzaysal verinin çok büyük miktarlarda olması bu verilerin işlenmesini çok pahalı ve uzaysal verinin detaylıca incelenmesini çok zor hale getirir. Kümeleme uzaysal verinin anlaşılmasını ve sürecin analizinin yapılmasını otomatik hale getirebilir.

Geniş uzaysal veri tabanlarında var olabilen ilginç karakteristikleri ve bireyleri çıkarmada ve özdeşleştirmede kullanılabilir.

Web Madenciliği: Burada kümeleme web üzerindeki belgelerin anlamlı gruplarını keşfetmede kullanılabilir. Web belgelerinin bu sınıflandırması, bilgi keşfinde yardımcı olur.

Genel olarak kümeleme, sınıflandırma gibi bulunan kümelerin işlenmesini sağlayan, diğer algoritmalarından önceki süreçtir.

2.3 Kümeleme Algoritmalarının Kategorileri

Literatürde önerilen çok fazla kümeleme algoritması mevcuttur. Bu kümeleme algoritmaları aşağıdaki gibi sınıflandırılabilir.

- Algoritmaya veri girişinin çeşidine göre.
- Veri noktaları arasındaki benzerliği tanımlayan kümeleme kriterine göre.
- Dayandığı teori ve esaslara göre (Bulanık Teori, İstatistik gibi).

Böylece kümeleri tanımlamak için kabul edilen metoda göre, algoritmalar aşağıdaki gibi sınıflandırılabilir (HALKIDI vd , 2001).

Hiyerarşik kümeleme: Hiyerarşik kümeleme, küçük kümeleri büyük bir kümede birleştirme veya büyük kümeleri küçük kümelere parçalama işlemini başarıyla yürütür. Bu algoritmanın sonucu dendogram olarak adlandırılan ve kümelerin birbirleriyle olan ilişkilerini gösteren, ağaç şeklinde yapıdan oluşur. Dendogram istenilen seviyede kesilerek, veri parçalarının ayrı gruplara bir kümelemesi elde edilebilir.

Bölümlemeli Kümeleme: Bölümlemeli kümeleme, veri setini direk olarak ayrı kümelerin bir setine ayrıştırmayı dener. Daha çok kriter fonksiyonunu optimize edecek bölünmelerin sayısını belirlemeye çalışır. Kriter fonksiyonu, verinin yerel veya global yapısı üzerinde durur ve bu kriter fonksiyonunun optimizasyonu dögüsel bir prosedür ile yapılır.

Yoğunluğa Dayalı Kümeleme: Bu çeşit kümelemedeki ana fikir, veri setindeki nesnelere yakınlıklarına göre yoğunluk koşullarına dayanan kümelere ayırmaktır.

Izgara Tabanlı Kümeleme: Bu çeşit algoritma, uzaysal veri madenciliğinde önerilir. Bu kümelemenin ana karakteristiği, herhangi bir aralığı sonlu sayıda hücreye bölmektir ve bütün işlemleri bu bölünen aralık üzerinde yapmaktır.

Bulanık kümeleme: Verileri kümelemek için bulanık teknikleri kullanırlar ve bu tekniklerde bir nesne birden fazla kümeye sınıflandırılabilir. Bu tip algoritmalar gerçek sayıların belirsizliğini ele aldığından günlük yaşamın tecrübelerine uygun kümeleme şekillerinin ortaya çıkmasını sağlar. En önemli bulanık kümeleme algoritması Bulanık C-Ortalamalar (BCO) algoritmasıdır.

Yukarıdaki kategorilerin her biri için, kümeleri bulmanın farklı algoritmaları mevcuttur. Böylece, değişken tiplerine göre veri seti aşağıdaki şekilde sınıflandırılabilir.

İstatistiksel, istatistiksel analiz kavramlarına dayanır. Bunlar nesnelere bölümlenmek için benzerlik ölçülerini kullanırlar ve sayısal veriler ile sınırlandırılırlar.

Kavramsal, kesin (koşulsuz) verileri kümelemek için kullanılır. Bu teknikler nesnelere taşıdıkları anlama göre kümeler.

Klasik kümeleme, tekniğinde bir veri noktasının sadece bir kümeye girdiği varsayılır. Bir verinin birden fazla kümeye girme durumu bu teknikte yoktur. Sonucu klasik küme olan kümeleme algoritmalarının çoğu, klasik kümeleme sınıfına girebilir.

Kohonen Ağ Kümeleme, sinir ağlarının kavramlarına dayanır. Kohonen ağı girdi ve çıktı düğümlerine sahiptir. Girdi katmanda kayıtnın her özelliği için bir düğümü vardır ve her biri her bir çıktı düğüme bağlanır. Her bağlantı çıktı düğüme karşılık gelen pozisyonu belirleyen bir ağırlık ile birleştirilir. Böylece algoritmaya göre gereği gibi değişen ağırlıklar çıktı düğümü bir kümeye taşır.

Genellikle, kümeleme algoritmaları verilen bir bölünmenin kalitesini ölçmek için bir kritere dayanır. Kümeleme algoritmaları girdi olarak bazı parametreler alır ve verilen parametreler için veri setinin en iyi bölünmesini tanımlamaya çalışır. Böylece, bu algoritmalar varsayımlara dayanan veri setinin bölünmesini tanımlar ve veri setini en iyi bölünmeye uydurur.

Kümeleme algoritmaları, önceliği bilinmeyen kümeleri keşfettiğinde, veri setinin sonuç bölünmeleri, çoğu uygulamada bazı değerlendirmeler gerektirir.

Örneğin “Veri setinde kaç tane küme var?”, “Meydana gelen kümelerin düzeni bizim verilerimize uyuyor mu?”, “Bizim verilerimiz için en iyi bölünme var mı?” şeklindeki sorular, küme sonuçlarının geçerliğini görmek amacıyla sorulur. Kümeleme algoritmalarının sonuçlarının nicel değerlendirilmesini yapmayı amaçlayan metotlar “Kümeleme Doğruluma” metotları olarak bilinirler (HALKIDI vd, 2001).

2.4 Kümeleme Yaklaşımları

2.4.1 Hiyerarşik kümeleme yaklaşımı

Hiyerarşik kümeleme, dendogram olarak bilinen kümelerin ağaç yapısını veya diğer bir deyişle bir küme hiyerarşisi ortaya çıkarır. Her küme düğümü, ortak ebeveynlere sahip noktaların çocuk ve kardeş küme bölünmelerini içerir. Böyle bir yaklaşım, farklı seviyelerdeki düğümlerin dikkatle incelenmesine olanak sağlar. Hiyerarşik kümeleme metotları toplanmış (agglomerative) ve bölen (divisive) metotları olarak sınıflandırılabilir (Jain vd, 1988, Kaufman vd, 1990). Toplanmış kümeleme tek noktadan (singleton) oluşan kümelere başlar ve bu kümeleri tekrarlı olarak en uygun kümeleri şekilde 2 veya daha fazla kümede birleştirir. Bölen kümeleme ise bütün veri noktalarını içeren tek bir kümeden başlar ve bu kümeleri en uygun biçimde tekrarlı olarak parçalara ayırır. Bu süreç bitirme kriteri sağlanana kadar devam eder.

Hiyerarşik kümelemenin avantajları aşağıdaki gibi sıralanabilir:

- Düğümlerin seviyesine ilişkin esneklik özelliğinin olması
- Benzerlik ve uzaklık biçimlerini ele almanın kolaylığı
- Çeşitli niteliklere uygulanabilme özelliğinin olması

Hiyerarşik kümelemenin dezavantajları ise şu şekildedir:

- Bitirme kriterinin belirsizliği
- Gerçekte çoğu hiyerarşik kümeleme algoritması, geliştirilme amaçlarına göre arada bulunan kümelere (dendogramda) tekrar ulaşılmasına izin vermez.

Hiyerarşik kümelemeye klasik yaklaşımlar, **Bağlantılı-Ölçülü** (Linkage Metric) olarak adlandırılan alt bölümde gösterilir. Hiyerarşik kümelemede, düzenli

özelliklerden oluşan veri noktalarının temsili bazen ikinci sırada gelmektedir. Onun yerine, hiyerarşik kümeleme genellikle, eğitim noktalarının arasındaki benzerlik veya uzaklık matrisleriyle ilgilenir. Bu matris bazen **bağlantısallık matrisi** olarak adlandırılır. Bağlantı ölçümleri bu matrisin elemanlarından oluşturulur. Bu kadar büyük bir matrisi bellekte tutma gereksinimi gerçekçi değildir. Bağlantısallık matrisi ile veri noktaları X, ağırlıkları veri çiftleri olan E kenarlarından oluşan $G=(X,E)$ bağlantısallık grafiği birleştirilebilir. Böylece hiyerarşik kümeleme ile grafik bölünmesi arasında bağlantı kurulur.

2.4.1.1 Toplanmış hiyerarşik kümeleme yöntemleri

Başlangıçta tüm birimlerin ayrı küme oluşturduğunu kabul ederek, n birimi hiyerarşik olarak sırasıyla $n, n-1, \dots, n-r, \dots, 3, 2, 1$ kümeye yerleştirmeyi amaçlayan bir yaklaşımdır.

Toplanmış (agglomerative) hiyerarşik kümeleme yönteminde, kümeleme sürecinin başlangıcında her birey bir kümedir, süreç sonunda ise tüm bireyler tek bir kümede toplanır (Tatlıdil, 2002). İşleyiş daha ayrıntılı bir şekilde aşağıdaki gibi bir algoritma ile ifade edilebilir.

1. n tane birey, n tane küme olmak üzere işleme başlanır.
2. En yakın (d_{ij} değeri en küçük olan) iki küme birleştirilir.
3. Küme sayısı bir azaltılarak yinelenmiş uzaklıklar matrisi bulunur.
4. 2 ve 3 no'lu adımlar n-1 kez tekrarlanır (Tatlıdil,2002).

Bu süreçte birden çok gözlemler kümenin vektör olarak gösterilebilmesi amacıyla değişkenlerin ortalama değerlerinden yeni vektör oluşturulmakta ya da bu kümedeki tüm gözlemler ile başka kümedeki gözlemlerin uzaklık ortalamaları kullanılabilir.

Yukarıda verilen algoritmaya dayalı yedi farklı hiyerarşik teknikten söz edilmektedir. Bu teknikler sırasıyla tek bağlantılı, tam bağlantılı, grup ortalama, merkezi, ortanca, minimum varyans ve Ward teknikleridir.

Toplanmış hiyerarşik kümeleme yöntemlerinde, birimlerin birbirleri ile birleştirilmesinde değişik yaklaşımlar uygulanmaktadır. Bu nedenle değişik isimlerle

anılan toplanmış hiyerarşik kümeleme yöntemleri bulunmaktadır. Bu yöntemlerden sıklıkla kullanılan ve genel kabul görmüş olanları aşağıdaki gibi sayılabilir (Özdamar, 2002).

(1) Tek bağlantılı kümeleme yöntemi (TekBKY)

En Yakın Komşuluk (Nearest Neighbor-Single Linkage-SLINK) olarak da bilinen bu teknikte uzaklıklar matrisi kullanılarak birbirine en yakın birey ya da kümeler birleştirilmekte ve birleştirme ardı ardına tekrarlanarak sürdürülmektedir (Tatlıdil, 2002).

Johnson tarafından önerilen bu teknikte eğer i ve j . küme birleştirilmiş ise birleştirilen kümenin k . küme ile ilişkisi uzaklık ölçütü

$$d_{k(i,j)} = \text{Min}(d_{ki}, d_{kj}) \quad (2.1)$$

Bu yöntem hiyerarşik kümeleme yöntemleri içinde bireylerin oluşturdukları hiyerarşik kümeleri belirlemede kullanılan en basit yöntemlerden biridir.

TekBKY ile hiyerarşik kümeleme yapmak için aşağıdaki işlem sırası izlenir:

- 1) X veri matrisinin D Öklid uzaklık matrisi hesaplanır.
- 2) Eğer istenirse D matrisinden Sim (benzerlik) matrisi hesaplanır.
- 3) D ya da Sim matrisinde en küçük değerli birimler hiyerarşik olarak birbirleri ile birleştirilir. Küme (ij) oluşturulur. i .ve j kümeleri birbirleri ile birleştirildikten sonra D (ya da Sim) matrisinde j . kümeye ilişkin satır silinir ve ij kümesinin uzaklığı (ya da benzerliği) i . kümenin uzaklığı olarak kalır ($d_i < d_j$).
- 4) Tüm kümeler birbirleri ile birleştirilinceye kadar 3. satırdaki işlemler tekrarlanır (Özdamar, 2002).

(2) Ortalama bağlantılı kümeleme yöntemi (OrtBKY)

OrtBK yönteminde bir birimin m . küme olarak hangi birim ya da kümelerle birleştirileceği, birimlerin yeni oluşan kümelerle olan uzaklıkları dikkate alınarak belirlenir. m . kümenin daha önce oluşan k . ve l . kümelerden hangisi ile birleşerek oluşacağını belirlemek için j . küme ile k . l . kümelerin uzaklıklarına bakılır. Bu

uzaklıklar k ve l kümelerinin eleman sayısı ile çarpılarak ağırlıklandırılır. Elde edilen toplam yeni oluşacak m . küme eleman sayısına bölünür (Özdamar, 2002).

m . kümenin j . küme ile olan uzaklığı (d_{mj});

$$d_{mj} = (N_k d_{kj} + N_l d_{lj}) / N_m \quad (2.2)$$

şeklinde belirlenir.

Birbirine en çok benzeyen çiftin bulunmasında grup içi benzerlik ortalaması k ve l kümelerine ait çiftlerin benzerlik ölçülerinden ve birim sayılarından yararlanılarak hesaplanır (Özdamar, 2002).

(3) Tam bağlantılı kümeleme yöntemi (TBKY)

En Uzak Komşuluk (Furhest Neighbor Complete Linkage-CLINK) olarak da bilinen bu teknik yine Johnson tarafından önerilmiştir. Tek bağlantılı tekniğine çok benzeyen bu teknikte tek farklılık iki küme arasındaki uzaklık olarak her kümedeki eleman çiftleri arasındaki uzaklığın en büyüğü almaktadır. Bu farklılığı yukarıdaki eşitliğe paralel olarak

$$d_{k(i,j)} = \text{Max}(d_{ki}, d_{kj}) \quad (2.3)$$

biçiminde gösterebiliriz (Tatlıldil, 2002).

Hala kullanılmakta olan bilgisayar algoritmalarının büyük çoğunluğu tek ve tam bağlantılı tekniklerini kullanmakla birlikte, tek bağlantılı tekniği sağlıklı sonuçlar vermesi açısından tercih edilir ancak işlemlerin uzun sürmesi açısından sakıncalıdır. Tam bağlantılı tekniği ise aynı küme içerisindeki bireylerin uzaklıklarının belli bir değerden küçük olması durumunda tüm kümelerin sağlıklı oluşturulmasını garanti etmemektedir. Son yıllarda sıkça kullanılmaya başlayan ortalama bağlantılı tekniği, bu iki uç teknik arasında sonuçlar vermesi nedeniyle bir alternatif olarak önerilmektedir. Ayrıca merkezi, ortanca ve minimum varyans teknikleri de bu üç tekniğin benzerleri olup uygulamada pek sık kullanılmamaktadır (Tatlıldil, 2002).

Algoritma:

Adım 1: Her bireyi kendi kümesine yerleştir. Tüm sırasız birey çiftleri için bireyler arasındaki uzaklığın listesini oluştur ve bu listeyi küçükten büyüğe doğru sırala.

Adım 2: Birbirlerine en uzak (uzaklık değerleri en büyük) olan birey ya da kümeler birleştirilir.

Adım 3: Bütün bireyler birleştirilen bir kümenin üyesi ise işlem bitirilir. Aksi takdirde Adım 2'ye geri dönülür.

Bağlantı kümelemeye dayanan hiyerarşik kümeleme, zaman karmaşıklığı problemine maruz kalır. Zaman karmaşıklığı problemine rağmen bu algoritmalar yaygın olarak kullanılır.

(4) McQuity bağlantılı kümeleme yöntemi (MBKY)

m. kümenin oluşumu, k. ve l. kümelerin j. küme ile olan uzaklıkları ortalaması dikkate alınarak belirlenir. Ağırlıksız ortalama bağlantı yöntemi olarak da bilinmektedir. Yeni oluşan m. ve j. kümeler arasındaki uzaklık;

$$d_{mj} = (d_{kj} + d_{lj}) / 2 \quad (2.4)$$

biçimde belirlenir (Özdamar, 2002).

(5) Merkezsel bağlantılı kümeleme yöntemi (MerBKY)

OrtBK yönteminin özel bir biçimindedir. m kümesinin j kümesine olan uzaklığı;

$$D_{mj} = (N_k d_{kj} + N_l d_{lj}) / N_m - N_k N_l d_{kl} / N_m^2 \quad (2.5)$$

biçimde belirlenir.

(6) Ortanca bağlantılı kümeleme yöntemi (OBKY)

McQuity bağlantı kümeleme yönteminin özel bir biçimidir. m ve j kümeleri arasındaki uzaklık;

$$d_{mj} = (d_{kj} + d_{lj}) / 2 - d_{kl} / 4 \quad (2.6)$$

biçiminde belirlenir (Özdamar, 2002).

(7) Ward bağlantılı kümeleme yöntemi (WBKY)

Ward bağlantı yöntemi merkezsel ve ortanca bağlantı kümeleme yöntemlerinin karma ve ağırlıklı biçimidir. m ve j kümeleri arasındaki uzaklık;

$$d_{mj} = ((N_j + N_k)d_{kj} + (N_j + N_l)d_{lj} - N_j d_{kl}) / (N_j + N_m) \quad (2.7)$$

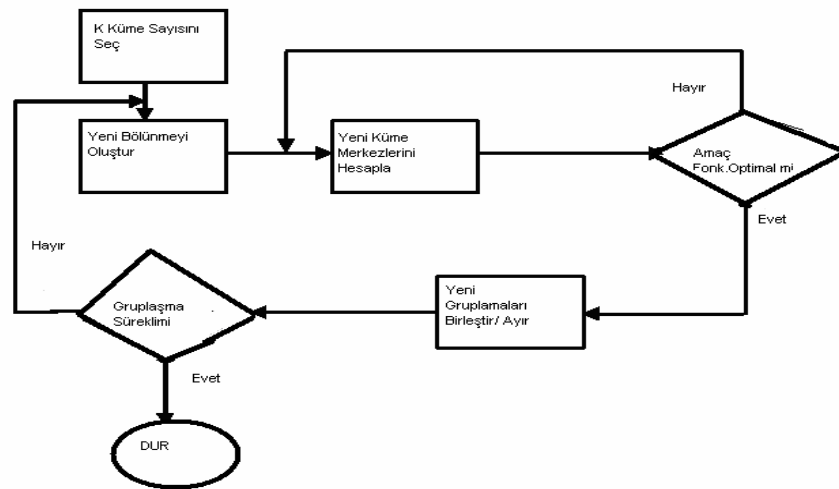
biçiminde belirlenir (Özdamar, 2002).

2.4.1.2 Bölen kümeleme yöntemleri

Bölen (divisive) kümeleme yöntemi, başlangıçta tüm birimlerin bir küme oluşturduğu kabul ederek birimleri hiyerarşik olarak n birimi sırasıyla 1,2,3,...,n-r,...,n-1,n kümeye ayırmayı amaçlayan bir yaklaşımdır. Toplanmış hiyerarşik kümeleme yönteminin tersidir. Toplanmış yönteme ilişkin sonuçlardan bölen yönteme ilişkin sonuçlarda elde edilebilir.

2.4.2 Bölümlemeli kümeleme yaklaşımı

Bölümlemeli kümelemenin temeli, başlangıç bölünmesinden başlayarak, bireyleri yinelemeli olarak kümeleme kriterini azaltacak şekilde kümelere tahsis etmek oluşturur. Yinelemeli bölümlemeli kümelemenin genel algoritması aşağıdaki şekilde görüldüğü gibidir.



Şekil 2.2 Bölümlemeli Kümelemenin Akış Diyagramı

Bölümlemeli kümeleme algoritmalarının yapılabilmesi için birçok kriter vardır ve bu kriterler şu şekildedir;

- Başlangıç Koşulları
- Küme Temsil Şeması
- Tahsis etme fonksiyonları
- En iyilik Kriteri
- Birleştir/Ayır Koşulları

Bölümlemeli kümeleme algoritmaları, yinelemeli yığın yordamı olduğundan dolayı, verilen başlangıç koşulları için global minimumu garanti etmez. Eğer uygulama alanı ile ilgili ön bir bilgi varsa, bu bilgi daha uygun başlangıç koşullarını belirlemek için kullanılabilir. Bununla birlikte özellikleri bilinmeyen bir veri kümesi için, bu metot, çok sayıda çalıştırılır ve en sık meydana gelen küme yapısı doğru bölünme olarak kabul edilir.

Bir küme temsili, genellikle küme içindeki bireyleri karakterize eden matematiksel veya geometrik bir yapıdır (Michalski vd, 1981). Mümkün temsil şemaları arasında; kümenin merkezi, kümenin en uçtaki üç bireyi, normal dağılım fonksiyonu, sınıflandırma ağacı, bağlayıcı ifadeler yer alır. Verilen bir küme temsil şeması, küme temsil fonksiyonu, verilen küme için en iyi temsili belirler. Tahsis etme fonksiyonları, temsil etme fonksiyonlarının tersidir: bir küme temsili verildiği zaman her bir kümeye giren bireyleri belirler.

Küme en iyilik kriteri, kümelerin doğruluğunu belirleyen bir ölçüdür. Bu kriter ayrıca yinelemenin devam edip etmeyeceğini belirtir. Küme en iyilik kriteri bu kümeleme algoritması için oldukça önemlidir. Doğru seçilmediği durumda yanlış kümelerin oluşmasına sebep olacaktır.

Bölümlemeli kümeleme algoritmaları, küme temsilcilerinin nasıl oluşturulacağına bağlı olarak, yinelemeli en iyileme bölünme algoritmaları k-medoids ve k-ortalamar olmak üzere iki gruba ayrılabilir.

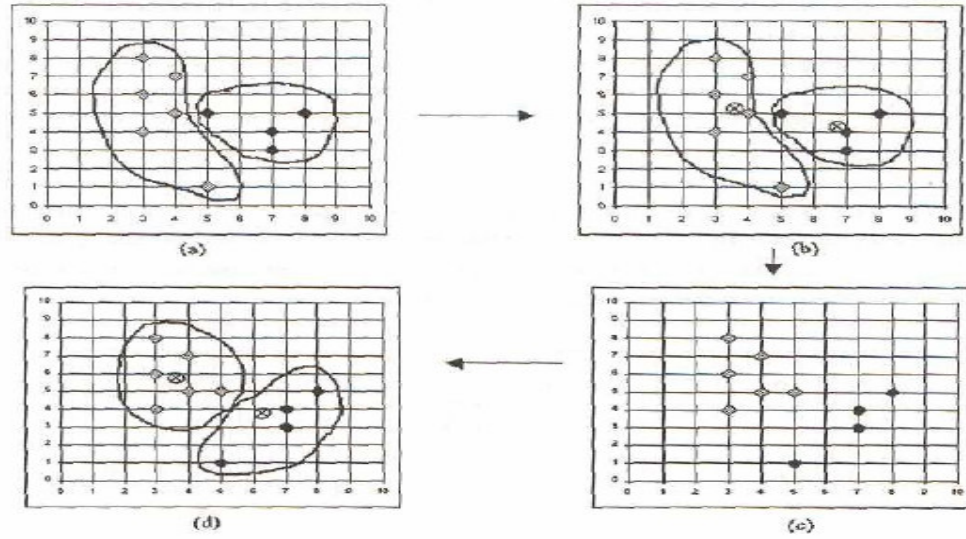
2.4.2.1 Klasik k-ortalamlar algoritması (KOA)

K-ortalamlar algoritması (KOA), 1975 yılında Hartigan tarafından tanımlanan standart kümeleme algoritmasıdır ve hala geniş çaplı kullanılmaktadır. K-ortalamlar algoritması, endüstriyel ve bilimsel anlamda kullanılan en popüler kümeleme araçlarından biridir. Adı, kitle merkezi (centroid) olarak adlandırılan, k tane kümenin her birini temsil eden noktaların ortalamasından gelir. Kesin niteliklerde çok daha iyi çalışmayacağı bellidir. Sayısal nitelikler için iyi bir geometrik ve istatistiksel anlama sahiptir. Nokta ile kitle arasındaki farklılıkların toplamı amaç fonksiyonu olarak kullanılan uygun aşağıdaki uzaklık ölçüsü ile ifade edilir.

$$E(C) = \sum_{j=1}^k \sum_{x_i \in C_j} \|x_i - c_j\|^2 \quad (2.8)$$

Burada x_i : veri noktalarını, c_j : merkezini k: küme sayısını göstermektedir. K-ortalamlar yöntemi, ilk önce n adet nesneden rasgele k adet nesne seçer ve bu nesnelerin her biri, bir kümenin merkezini veya orta noktasını temsil eder. Geriye kalan nesnelere her biri kendisine en yakın olan küme merkezine göre kümelere dağılırlar. Yani bir nesne hangi kümenin merkezine daha yakın ise o kümeye yerleşir. Ardından her küme için ortalama hesaplanır ve hesaplanan bu değer o kümenin yeni merkezi olur. Bu işlem tüm nesnelere kümelere yerleşinceye kadar devam eder (Han, 2000). Bir nesne grubunun, Şekil 2.3'de görüldüğü gibi uzayda konumlanmış olduğu varsayalım. Kullanıcının bu nesnelere iki kümeye ayırmak istediği varsayılırsa, $k=2$ olur (Han, 2000). Şekil 2.3'te uzayda ilk önce rasgele iki nesne, iki kümenin merkezi olarak seçilmiş ve diğer nesnelere de bu merkezlere olan yakınlıklarına göre iki kümeye ayrılmışlardır. Bu ayrıma göre her iki kümenin nesnelere yeni ortalaması alınmış ve bu değer kümelerin yeni merkezleri olmuştur. Bu yeni merkezler Şekil 2.3(b)'de üstünde çarpı işareti bulunan noktalarla gösterilmektedir. Bu yeni çarpı işaretli merkezlere göre, her iki kümede de birer nesne diğer kümenin merkezine daha yakın duruma gelmişlerdir. Bu durum Şekil 2.3(c)'de görülmektedir. (5,1) koordinatındaki nesne ile (5,5) koordinatındaki nesne küme değiştirmişlerdir. Her iki kümedeki bu yeni katılımlar ile kümelerdeki nesnelere yeni ortalamaları ve dolayısıyla merkezleri değişmiştir (Han, 2000). Yeni

hesaplanan merkezler Şekil 2.3(d)'de üstünde çarpı işareti bulunan noktalarla gösterilmektedir. Artık açıkta bir nesne kalmadığı ve her nesne içinde bulunduğu kümenin merkezine en yakın durumda bulunduğu için k-ortalamlar yöntemi ile kümelere bölünme işlemi Şekil 2.3(d)'de görüldüğü gibi sonlanmıştır (Han,2000).



Şekil 2.3. k-ortalamlar Yöntemi

k-ortalamlar yöntemi, sadece kümenin ortalaması tanımlanabildiği durumlarda kullanılabilir (URL7). Kullanıcıların k değerini, yani oluşacak küme sayısını belirtme gerekliliği bir dezavantaj olarak görülebilir. Esas önemli olan dezavantaj ise dışarıda kalanlar (outliers) olarak adlandırılan nesnelere karşı olan duyarlılıktır (Han, 2000). Değeri çok büyük olan bir nesne, dâhil olacağı kümenin ortalamasını ve merkez noktasını büyük bir derecede değiştirebilir. Bu değişiklik kümenin hassasiyetini bozabilir (URL6).

k-ortalamlar algoritması, varyans analizinin sağlam temeli üzerine kurulmuştur. K-ortalamlar algoritması ile ilgili olağan bazı şüpheler vardır.

- Sonuçlar küme merkezlerinin başlangıç tahminlerine bağlıdır.
- Küme sayısı k ne olacağı çok açık değildir.
- Bu işlem aykırı değerlerin olması durumunda başarılı sonuçlar vermez.

- Bu algoritma ölçeklenebilirlik özelliğinden yoksundur.
- Sadece sayısal nitelikleri kapsar.
- Meydana gelen kümeler dengeli olmayabilir (HALKIDI, 2001).

Başlangıç kümelerinin etkilerini azaltmak için basit birkaç yol vardır. İlk olarak, rasgele üretilen birkaç küçük veri örneği üzerinde k-ortalamlar hesaplanır. Bu oluşturulan sistemlerden her biri, tüm örneklerin birleşimi için başlangıç olarak kullanılır. Bu yolla oluşturan en iyi sistemin kitle merkezi, bütün veriler üzerinde k-ortalamlar algoritmasının uygulamak için başlangıç tahmini olarak önerilir.

Algoritma;

Adım 1: Algoritmanın başlangıç değerleri için k tane kitle merkezi üret.

Adım 2: (2.8) ifadesini en küçükleyecek şekilde her bir bireyi kümelere tahsis et.

Adım 3: Üyelik değerlerini güncelle.

Adım 4: her bir i bireyi için

$$h = \operatorname{argmin}_{r \neq j} \frac{n_r \|x_i - c_r\|}{n_r - 1} \quad (2.9)$$

Burada n_r r. kümeye giren eleman sayısı. Eğer $\frac{n_h \|x_i - c_h\|}{n_h - 1} < \frac{n_j \|x_i - c_j\|}{n_j - 1}$ ise i. bireyi j. kümeden h. kümeye taşı

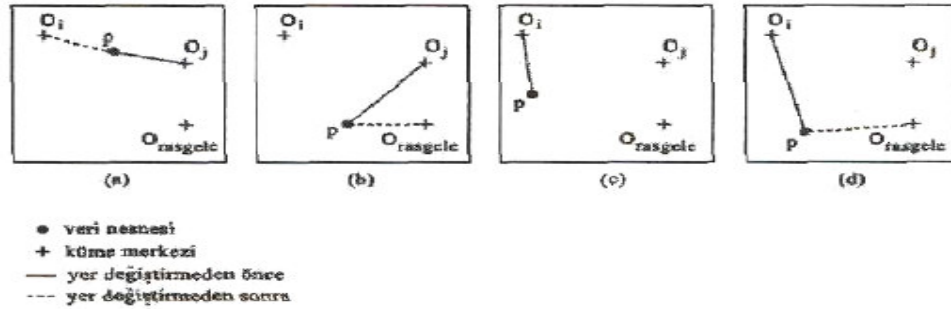
Adım 5: Eğer herhangi bir bireyin yeri değişirse Adım 4'e geri dön, aksi takdirde dur.

K-ortalamlar algoritmasının en büyük avantajı zaman karmaşıklığının O(n) yani küçük olmasıdır.

2.4.2.2 Klasik k-medoid algoritması (KMA)

Değeri çok büyük olan bir nesne, dâhil olacağı kümenin ortalamasını ve merkez noktasını büyük bir ölçüde değiştirebildiğinden kümedeki nesnelerin ortalamasını almak yerine, kümede ortaya en yakın noktada konumlanmış olan nesne anlamındaki medoid kullanılabilir. Bu işlem k-medoids yöntemi ile gerçekleştirilir. k-medoids

kümeleme yönteminin temel stratejisi ilk olarak n adet nesnede, merkezi temsili bir medoid olan k adet küme bulmaktır (Han, 2001). Geriye kalan nesnelere, kendilerine en yakın olan medoide göre k adet kümeye yerleşirler. Bu bölünmelerin ardından kümenin ortasına en yakın olan nesneyi bulmak için medoid, medoid olmayan her nesne ile yer değiştirir. Bu işlem en verimli medoid bulunana kadar devam eder (Han, 2001). Şekil 2.4'de O_i ve O_j iki ayrı kümenin medoid'lerini, O rasgele seçilen ve medoid adayı olan bir nesneyi, p ise medoid olmayan bir nesneyi temsil etmektedir. Şekil 2.4 O rasgele'nin, şu anda medoid olan O_j 'nin yerine geçip, yeni medoid olup olamayacağını belirleyen dört durumu göstermektedir (Han, 2001).



Şekil 2.4 k-medoids Yöntemi

- (a): p nesnesi su anda O_j medoidine bağlıdır (O_j medoidinin bulunduğu kümededir). Eğer O_j , O rasgele ile yer değiştirir ve p O_i 'ye en yakınsa, p nesnesi O_i 'ye geçer.
- (b): p nesnesi su anda O_j medoidine bağlıdır. Eğer O_j , O rasgele ile yer değiştirir ve p O rasgele'ye en yakınsa, p nesnesi O rasgele'ye geçer.
- (c): p nesnesi su anda O_i medoid'ine bağlıdır. Eğer O_j , O rasgele ile yer değiştirir ve p hala O rasgele 'ye en yakınsa, p nesnesi yine O_i 'ye bağlı kalır.
- (d): p nesnesi su anda O_i medoid'ine bağlıdır. Eğer O_j , O rasgele ile yer değiştirir ve P O rasgele 'ye en yakınsa, p nesnesi O rasgele 'ye geçer.

Algoritma

Adım1: Veriler arasından medoid olarak rasgele k tane birey seç.

Adım2: $i \in$ seçilmiş bireyler (medoid olarak), $h \in$ seçilmemiş bireyler bunların yerlerinin değiştirildiğini varsayalım. i ve h arasındaki uzaklık $d(x_i, x_h)$ ile gösterilirse,

Toplam yer değiştirme katkısı T_{ih} şöyle hesaplanır.

$$T_{ih} = \sum_j C_{jih} \quad (2.10)$$

Burada C_{jik} j. birey için i 'nin h'a katkısıdır ve aşağıdaki gibi 4 farklı şekilde hesaplanabilir.

-- j. bireyin i. medoid tarafından tanımlanan kümeye girdiğini ve j. ve h. bireyler arasındaki uzaklığın $d(x_j, x_h)$ olduğunu varsayalım. Bu durumda; eğer h j'ye ikinci en iyi medoid i' sinden daha uzaksa

$$C_{jih} = d(x_j - x_{i'}) - d(x_j - x_i) \quad (2.11)$$

Eğer h j'ye i' 'den daha yakınsa

$$C_{jih} = d(x_j - x_h) - d(x_j - x_i) \quad (2.12)$$

--Eğer j k. kümeye giriyorsa $k \neq i$ j. birey ile h. birey arasındaki uzaklığın kontrol edilmesi gerekir.

--- Eğer h ile j arasındaki uzaklık k. medoid ile j arasındaki uzaklıktan daha büyükse o zaman j. bireyin katkısı

$$C_{jih} = 0 \quad (2.13)$$

--Eğer h ile j arasındaki uzaklık k. medoid'le j arasındaki uzaklıktan daha küçükse

o zaman j. bireyin katkısı

$$C_{jih} = d(x_j - x_h) - d(x_j - x_k) \quad (2.14)$$

Adım 3: $(i^*, h^*) = \arg \min_{i, h} T_{ih}$ $T_{ih} < 0$ ise i^* ile h^* yer değiştir.

Adım 2'ye geri dön.

Adım 4: Medoid olarak seçilmemiş bireyleri en yakın medoid'e göre kümelere yerleştir.

k-medoid yöntemi veri seti içindeki aykırı değerler olması durumunda da iyi çalışır.

Bu yöntemi kullanmanın dezavantajları ise şu şekildedir:

- Bu algoritmaya girdi değeri olarak k küme sayısının verilmesi gerekmektedir. Bu nedenle iyi bir kümeleme elde etmek için k sayısının ne olacağına karar vermek gerekir. Bu seçimin kullanıcıya bağlı olması bu yöntemin dezavantajıdır.
- Zaman karmaşıklığı, algoritmanın her yinelemesi için $O(n^2)$ 'dir.
- Medoid'ler kullanıldığından, kümelerin başka herhangi bir şekilde tanımlamasını sağlamaz.

2.4.3 Yoğunluğa dayalı kümeleme yaklaşımı

Yoğunluğa dayalı kümeleme (YDK), tipik olarak kümelere, veri seti içindeki bireylerin yoğun olduğu bölgeler gözüyle bakar ve bu veri setini düşük yoğunluktaki bölgelere bölmeye çalışır. Bu kümelemeye göre noktaları sonlu sayıda kümeye bölmek için, yoğunluk, bağlanabilirlik ve sınır gibi kavramların bilinmesi gerekir. Bu kavramların noktaların en yakın komşularıyla ilgili olduğu açıktır. Yoğunluğa dayalı kümeleme algoritmaları keyfi şekilli kümeleri ortaya çıkarmada başarılıdır. Bu tür kümeleri bulmak diğer birçok kümeleme algoritması için problem olurken, yoğunluğa dayalı kümeleme algoritmaları bu tür kümeleri rahatlıkla ele alır.

Bu yöntem, ayrıca aykırı gözlemlere karşı doğal bir koruma sağlar. Ancak bu gibi iyi özellikleri olmasına rağmen, bazı rahatsızlıklardan dolayı ölçülü olarak kullanılmaktadır.

Yoğunluğa dayalı kümeleme algoritmalarından yaygın olarak kullanılan algoritma, DBSCAN'dır. Bu algoritmanın ana fikri, küme içindeki her nokta (bu noktalara merkez gözüyle bakıldığında) için; verilen bir yarıçap etrafında en az minimum sayıda nokta olmak zorundadır. DBSCAN Ester tarafından önerilen artan (incremental) kümeleme algoritması gibi kullanılır. Yoğunluğa dayalı doğası

yüzünden bir bireyin silinmesi veya eklenmesi, sadece bu bireyin komşuları içindeki geçerli kümelemeyi etkiler ve böylece DBSCAN'a dayanan bu algoritmalar ile var olan kümelemeye ekleme ve silme işlemleri yapılabilir (Ester vd,1998).

Başka bir yoğunluğa dayalı kümeleme algoritması ise DENCLUE (Hinneburd vd, 1998) dır. Bu algoritma, büyük çoklu ortam veri tabanlarını kümelemek için kullanılan yeni bir yaklaşım sunar. Bu yaklaşımın temel amacı, veri noktalarının etki fonksiyonunun toplamı şeklinde noktaların yoğunluğunu analitik olarak modellemektir. Etki fonksiyonu, bir veri noktasının komşularının içinde etkisini tanımlayan bir fonksiyon olarak görülebilir. Sonra **yoğunluk çekiciler** (density attractor) belirlenerek kümeler teşhis edilebilir. Yoğunluk çekiciler, toplam yoğunluk fonksiyonunun yerel maksimumudur. Ek olarak, keyfi şekilli kümeler toplam yoğunluk fonksiyonuna dayanan basit bir eşitlik tarafından kolayca tanımlanabilir. DENCLUE algoritmasının ana avantajı, aykırı değerlerin çok miktarda olduğu veri setlerinde özelliklerinin iyi bir şekilde kümelenemesine ve çok boyutlu veri setlerindeki keyfi şekilli kümelerin matematiksel olarak tanımlanmasına olanak sağlamasıdır. Ancak, DENCLUE kümeleme iki tane parametreye dayanır ve kümeleme sonuçlarının kalitesi bu parametrelerin seçimine bağlıdır. Bu parametreler:

- i) Bir veri noktasının komşularının içindeki etkisini belirleyen N .
- ii) Yoğunluk çekicilerin sayısını azaltmaya izin veren, performansı geliştirmeye yardımcı olan ve yoğunluk çekicilerin anlamlı olup olmadığını tanımlayan ϵ dır.

2.4.4 Izgara tabanlı kümeleme yaklaşımı

Izgara tabanlı kümeleme (ITK) algoritmaları, veri uzayını sonlu sayıda hücreler ile niceliklendirir ve bütün işlemleri bu nicelik uzayı üzerinde yapar. Tüm kümeleme yöntemleri, ızgara yapısı üzerinden hesaplanır. Bu yaklaşımın ana avantajı, sadece nicelik uzayındaki her bir boyuttaki hücrelerin sayısına bağlı olan, veri sayısından bağımsız olan işleme zamanının hızlı olmasıdır. Izgara tabanlı algoritmalar 3 alt bölüme ayrılabilir.

STING (Statistical Information Grid) algoritması, ızgara tabanlı kümeleme yönteminin en iyi temsilcisidir ve hiyerarşik yapıyı kullanarak uzaysal bölgeyi dikdörtgen şeklindeki hücrelere böler. STING (Wang vd, 1997) algoritması, hücrelerin içindeki bireylerin sayısal özelliklerinin her birinin ortalama, standart

sapma, varyans, minimum, maksimum gibi istatistiksel parametrelerini hesaplar. Sonra farklı seviyelerdeki kümeleme bilgisini temsil ettiği gibi, ızgara hücrelerinin hiyerarşik yapısını oluşturur. STING, yeni bir bireyin kümelere verimli tahsisini veya sorgulama için kümeleme bilgisinin kullanımını kolaylaştırır. Bu algoritmanın diğer kümeleme yöntemlerinden farklı avantajları şu şekildedir:

- Veriler hakkındaki özet bilgiyi (küme merkezleri gibi) temsil eden her bir hücrede depolanan istatistiksel bilgiden dolayı, hesaplanması sorgulamadan bağımsızdır.
- Izgara yapısından dolayı paralel işlemeyi ve güncelleştirmeleri kolaylaştırır.
- Yeterlilik en büyük avantajıdır. Hücrelerin istatistiksel parametrelerini hesaplamak için ilk olarak veri tabanını oluşturur ve bu nedenle oluşan kümelerin zaman karmaşıklığı, n bireylerin toplam sayısı olmak üzere $O(n)$ 'dir. Hiyerarşik yapı oluştuktan sonra, sorgu işleme zamanı, $O(g)$ 'dir. Burada g , genellikle n 'den daha küçük olan en düşük seviyedeki ızgara hücrelerinin toplam sayısını gösterir.

WAVECLUSTER (Sheikholeslami vd, 1998), literatüre en son giren ızgara-tabanlı kümeleme algoritmasıdır. Bu algoritma, uzaysal veriyi frekans alanına dönüştürmek için ve dönüştürülen uzaydaki yoğun olan bölgeleri bulmak için işaret işleme tekniklerini kullanır. İlk olarak veri uzayına çok boyutlu ızgara yapısı yüklenerek veri özetlenir (Han vd, 2001). Her bir ızgara hücresi, hücrelerle eşleşen nokta grupları hakkındaki bilgiyi özetler. Sonra orijinal özellik uzayını dönüştürmek için, dalgacık dönüşümünü kullanır. Dalgacık dönüşümü uygulandığında, ayrışmanın farklı seviyelerinde bireyler arasındaki göreceli uzaklığı korumak için veriler üzerinde dönüşüm işlemi yapılır. Bu özellik veri seti içindeki doğal kümelerin daha ayırt edilebilir olmasını sağlar. Sonra, yeni etki alanındaki yoğun olan bölgeler araştırılarak kümeler belirlenebilir.

Bu yöntemin avantajları şu şekildedir:

- Bu kümeleme yöntemi danışmansız kümeleme sağlar ve noktaların yoğun olduğu bölgeleri ortaya çıkarmak için çizgi-biçimli (line-shaped) filtre kullanır. Bu özelliği sayesinde aynı zamanda küme sınırları dışındaki zayıf bilgilerin ortadan kalkmasını sağlar.

- Çoklu-ayırıştırma (multi-resolution) özelliği, farklı doğruluk seviyelerindeki kümeleri ortaya çıkarmada yardımcı olur.
- n veri tabanındaki bireylerin sayısı olmak üzere, zaman karmaşıklığı $O(n)$ 'dir
- Keyfi şekilli kümeleri ortaya çıkarmada ve büyük veri setlerini işlemede başarılıdır.
- Aykırı gözlemler olması durumunda da başarılı sonuçlar verir.
- Küme sayısı veya komşuluk yarıçapı gibi girdi parametrelerine gerek duymaz

CLIQUE kümeleme algoritması, maksimum boyutlu alt uzaylarda yoğun kümeleri teşhis eder. Bu algoritma orijinal uzaydaki alt uzayların içindeki kümeleri bulmanın daha etkili olduğunu kanıtlamaya çalışır, çünkü çok boyutlu veri uzayında aykırı ve eşit olarak dağılmış değerler olabilir. Çok boyutlu uzaylarda dahi, veri uzayında herhangi bir yerdeki noktaların ortalama yoğunluğunun çok düşük olması muhtemel bir durumdur. CLIQUE algoritması özellik uzayını hücrelere böler. Veri noktalarının yoğun olduğu bölgeleri bulmak için her bir hücre içindeki noktaların sayısını bulur (Yu vd, 1998).

2.4.5 Bulanık kümeleme (BK) yaklaşımı

Bulanık kümeleme yöntemi, kümeler birbirinden belirgin bir şekilde ayrılmıyorsa ya da üyeliklerinde bazı birimler küme üyeliğinde kararsızsa uygun bir yöntem olarak ortaya çıkmaktadır. Bulanık kümeler kümedeki birimin üyeliği olarak tanımlanan 0 ile 1 arasındaki her birimi belirleyen fonksiyonlardır. Birbirine çok benzeyen birimler aynı kümede yüksek üyelik ilişkisine göre yer alırlar. Bundan dolayı Bulanık Kümeleme Yöntemi, birimlerin kümeye ya da kümelere ait olabilme katsayılarını hesaplar. Üyelik katsayılarının toplamı daima 1'e eşittir. Böylelikle birim en yüksek üyelik katsayısına sahip olduğu kümeye atanır. Üyelik fonksiyonları, kümedeki elemanlar sürekli veya süreksiz olsun bir bulanık kümedeki bulanıklığı karakterize fonksiyonlardır. Klasik kümeleme yöntemlerinde ise her bir birim sıfır olmayan sadece bir üyelik katsayısına sahiptir ve bu değer daima 1 dir. Dolayısıyla klasik kesin kümeleme yöntemleri, bulanık çözümlemenin sınırlı bir durumudur (Bezdek vd, 1992, Kaufman vd, 1990).

2.5 Kümeleme Algoritmalarının Karşılaştırılması

Kümeleme işlemi birçok uygulama için kullanışlı bir araçtır. Birçok bilim dalındaki araştırmacılar kümeleme problemiyle karşılaşmaktadır. Ancak, çeşitli bilim dallarındaki kavramları (örneğin veritabanları, öğrenme makineleri, desen tanıma, istatistik) birleştirmek zor bir problemdir. Bu yüzden, varsayımlar arasındaki ve araştırmaların arasındaki farklılıklar, çok sayıda kümeleme yöntemlerin ortaya çıkmasına sebep olmuştur.

Bölümlemeli kümeleme algoritmaları, çoğunlukla sayısal veri setlerine uygulanırlar. Bu algoritmaların başında k-ortalamalar ve k-medoid'ler gelir. Bölümlemeli algoritmalarının en önemli özelliği aykırı değerlere karşı duyarlı olmalarıdır ve konveks şekilli olmayan kümeleri keşfetmek için uygun olmamalarıdır. Bundan başka bu algoritmalar veri setini bölmek için kesin varsayımlara dayanırlar (küme sayısı gibi). Bu kümeleme sürecinin sonucu, keşfedilen kümelerin temsili noktalarının kümesidir. Bu noktalar, kümeleme algoritmasına bağlı olarak orta noktalar veya medoid'ler (genellikle kümenin ortasında bulunan birey) olabilir. Kümeleme kriteri olarak, algoritmanın amacı kümeyi temsil eden noktalar ile küme içindeki noktalar arasındaki uzaklığı minimize etmektir.

Hiyerarşik kümeleme algoritmaları, dendogram olarak adlandırılan, veritabanının hiyerarşik ayrışımını yaratır. Bu algoritmalar aykırı gözlem olması durumunda bölümlemeli algoritmalara göre daha başarılı sonuçlar verir. Ancak bu algoritmaların lineer olmayan zaman karmaşıklığı problemi en büyük dezavantajıdır.

Diğer bir kümeleme yöntemi olan YDK ise keyfi şekilli noktaların kümelenmesi için oldukça uygundur. Bundan başka aykırı gözlemlerin olması durumunda da verimli çalışır. Bu kategorinin en yaygın kullanılan iki algoritması vardır: DBSCAN, DENCLUE. Bu algoritmalarda başlangıç giriş parametrelerine bağlı olarak çalışırlar. YDK algoritmalarının zaman karmaşıklığı $O(n \log n)$ 'dir.

ITK algoritmaları, veri uzayı için bir ızgara (grid) tanımlar ve bütün işlemleri bu nicelik uzayı üzerinde gerçekleştirir. Bu kümeleme algoritmaları genel anlamda büyük veri tabanları, keyfi şekilli kümelerin bulunması ve aykırı değerlerin olması durumunda verimlidir. STING, en iyi bilinen ITK algoritmalarından biridir. Bu

algoritma uzaysal bölgeyi, dikdörtgen şeklindeki hücelere böler ve bu hücrelerin içindeki bireylerin sayısal özelliklerinin istatistiksel parametrelerini hesaplar. Bu ızgara yapısı paralel işlemeyi kolaylaştırır.

BK algoritmaları, veri uzayındaki bireylerin farklı üyelik dereceleri ile her kümeye girmesine izin veren bölünme-optimizasyon teknikleridir. En yaygın kullanılan bulanık kümeleme algoritması Bulanık C-Ortalamlar (BCO) algoritmasıdır. Bu algoritma, klasik C-ortalamlar algoritmasının bulanık uygulamalar için genişletilmiş halidir (Bezdek vd, 1984). BCO algoritması, kümenin merkezi (center) olarak adlandırılan, her küme içindeki kümeyi en iyi temsil eden noktayı bulmaya çalışır. Ayrıca küme içindeki her bir birey bu kümeye aitlik derecesini gösteren bir değere sahiptir. Buradan anlaşılacağı gibi bulanık kümeleme algoritmalarının bu özelliği, her bir bireyin farklı derecelerle her kümeye ait olmasını sağlar.

2.6 Kümelemenin Doğrulaması

Kümeleme analizinin en önemli hususlarından biri de ilgilenilen veri setine ilişkin en iyi bölünmeyi bulmak için kümeleme sonuçlarının değerlendirilmesidir. Bunun ana konusu Kümelemenin Doğrulaması'dır. Bu bölümde, literatürde önerilen çeşitli kümeleme doğrulama yaklaşımlarına değinilecektir.

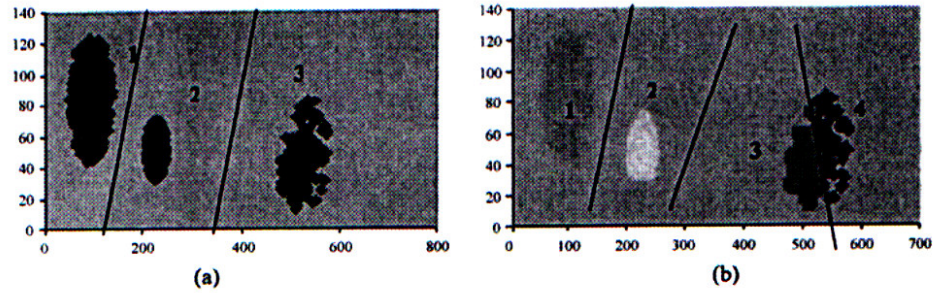
2.6.1 Problemin belirlenmesi (Problem Specification)

Kümeleme metotlarının amacı, veri seti içindeki önemli grupları keşfetmektir. Genellikle, kümeler için her birine en yakın olan üyeler araştırılır ve bu üyelere göre gruplara ayrılır. Burada önemli olan problem kümelerin en uygun sayısına karar vermektir. Çoğu algoritmanın deneysel değerlendirmelerinde, 2 boyutlu veri seti kullanılır. Öyle ki; okuyucu kümeleme doğruluğunu görsel olarak değerlendirebilir (veri setinin bölünmesinin ne kadar iyi olduğunu). Buradan kümelerinin görselliğinin kümeleme sonuçlarının doğrulanması üzerinde önemli bir etkisi olduğu açıktır. Ancak veri setinin çok boyutlu olması durumunda, veri setinin görsel etkilerini görmek zor olacaktır.

Çeşitli kümeleme algoritmaları, aşağıdaki durumlara bağlı olarak farklı davranırlar.

- **Veri Setinin Özellikleri:** Geometrisi ve kümelerin yoğunluk dağılımı gibi.

- **Girdi parametrelerinin deęerleri**



Şekil 2.5. (a) 3 küme (b) 4 küme k-ortalamlar'dan elde edilen sonuçlar.

Örneğin, Şekil 2.5 (a)'ya bakıldığında, verilen veri setinden, 3 tane küme keşfedilebileceği açıktır. Eğer bu veri seti K-ortalamlar algoritması ile 4 kümeye bölünmeye çalışılırsa o zaman Şekil 2.5 (b)'deki görüntü elde edilecektir. Ancak bu verilen veri seti için optimal bölünme değildir. Bu veri seti için optimal küme sayısı 3 olacaktır.

Sonuç olarak, eğer kümeleme algoritmasının parametreleri yanlış seçilirse, bu durumda kümeleme algoritmasının sonucu, veri seti hakkında yanlış kararlar alınmasını sağlayabilecek bir bölünme olabilir. Veri setini en iyi şekilde bölecek optimal küme sayısına karar vermek için, bazı araştırmacılar tarafından bir takım algoritmalar denenmiştir (Dave, 1996; Gath and Geva,1989;Rezaee vd 1998; Smyth 1996; Theodoridis ve Koutroubas, 1999; Xie ve Beni , 1991).

2.6.2 Kümeleme doğrulamanın temel kavramları

Bir kümeleme algoritmasının sonuçlarını değerlendirme prosedürü Kümeleme Doğrulama olarak bilinir. Genel anlamda, kümeleme doğruluğunu incelemek için 3 yaklaşım mevcuttur. Bunlardan ilki, **harici kriter** (External Criterion)'dir. Bu kriter, veri setinin ve küme yapısının önceden açıkça belirtilmesine dayanır. İkinci yaklaşım, **dahili kriter** (Internal Criterion) dayanır. Bu yaklaşım kullanıldığında, bir kümeleme algoritmasının sonuçları, sadece veri setinin özellikleri ve nicelikleri kullanılarak değerlendirilir. 3. kümeleme doğrulama yaklaşımı ise, **görelî kriter** (Relative Criterion)'dir. Bu yaklaşımda temel fikir,

kümeleme yapısını, aynı algoritmayı fakat farklı parametre değerlerini kullanan diğer kümeleme yapılarıyla karşılaştırmaktır. Kümelemeyi değerlendirmek için önerilen 2 tane kriter vardır (Berry, Linoff,1996).

1) Yoğunluk (Density): Her bir kümenin üyeleri, o kümeye diğer kümelere olduğundan mümkün olduğunca daha yakın olmalıdır. En yaygın kullanılan yoğunluk ölçüsü, minimize edilebilen varyanstır.

2) Ayırma (Separation): Kümelerin kendi aralarında geniş aralıklar olmalıdır. İki farklı küme arasındaki uzaklığı ölçen 3 yaygın yaklaşım bulunmaktadır.

- **Tek Bağlantı:** Kümelere en yakın üyeler arasındaki uzaklığı ölçer.
- **Tam Bağlantı:** En uzak üyeler arasındaki uzaklığı ölçer.
- **Küme Merkezlerinin Karşılaştırılması:** Küme merkezleri arasındaki uzaklığı ölçer.

İlk iki yaklaşım, istatistiksel testlere dayanır ve en büyük dezavantajları hesaplama zamanlarının çok olmasıdır. Bundan başka, bu yaklaşımlarla ilgili belirtiler, veri setinin önceden belirlenen şemaya ne kadar uyduğunun derecesini ölçmeyi amaçlar. Diğer taraftan, 3. yaklaşım bazı varsayımlar ve parametreler altında tanımlanabilen kümeleme algoritmasının en iyi kümeleme şemasını bulmaya çalışır.

2.6.3 Doğruluk indeksleri

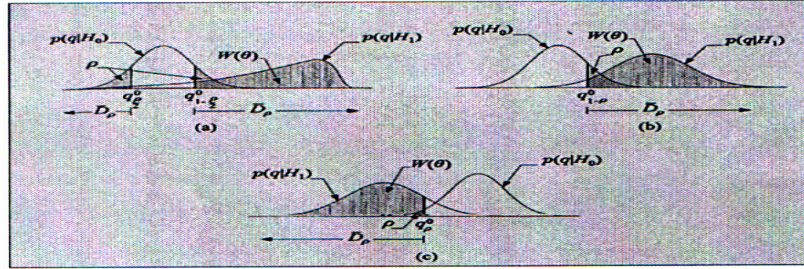
Bu bölümde, bilinen kümeleme doğrulama metotlarından kümeleme sonuçlarının değerlendirilmesini niceliksel olarak yapan metotlardan bahsedilmiştir.

2.6.3.1 Harici kriter

Bu yaklaşımın temel fikri, veri noktalarının rasgele yapılandırılıp yapılandırılmadığını test etmektir. Bu analiz *Sıfır Hipotezine* dayanır. H_0 veri setinin rasgele yapılandırıldığını ifade eder. Bu hipotezi test etmek için, hesaplama karmaşıklığına yol açabilecek istatistiksel testler kullanılır. Monte Carlo teknikleri ise bu yüksek hesaplama zamanı probleminin çözümü için kullanılır (Theodoridis vd, 1999).

➤ **Monte Carlo Kümeleme Doğrulamada Nasıl Kullanılır:** Monte Carlo tekniklerinin kullanımının amacı, tanımlanan istatistiksel göstergelerin olasılık

yoğunluk fonksiyonlarının hesaplanmasıdır. İlk olarak, çok miktarda yapay veri setleri oluşturulur. Bu yapay veri setlerinin her biri X_i olarak adlandırılır. Sonra q_i ile gösterilen, indeks değeri hesaplanır ve bu değerlerin birbirine göre grafiği çizilirse, bu grafik indeksin olasılık yoğunluk fonksiyonunun yaklaşığı olacaktır.



Şekil 2.6 (a) İki-yönlü indeks için güven aralığı (b)Sağ-Yönlü indeks (c) sol-yönlü indeks.

(Burada q_p^0 H_0 hipotezi altında q 'nin oranı ρ). Şekil 2.6'da görülen \bar{D}_p bölgeleri H_0 hipotezi için kabul edilemez bölgeleridir.

➤ **P bölünme için C Küme Yapısının Karşılaştırılması:** $C = \{C_1, \dots, C_m\}$ X veri seti için küme yapıları olduğunu varsayalım. $P = \{P_1, \dots, P_s\}$ verinin tanımlanan bölünmesi olsun. Aşağıdaki ifadeler kullanılarak (x_v, x_u) nokta çiftlerine gönderilebilir.

- 1) **SS:** Hem C kümeleme yapısının aynı kümelerine hem de P bölünmesinin aynı gruplarına giren noktalar ise
- 2) **SD:** C'nin aynı kümelerine ve P'nin farklı gruplarına giren noktalar ise
- 3) **DS:** C'nin farklı kümelerine ve P'nin aynı gruplarına giren noktalar ise
- 4) **DD:** C'nin farklı kümelerine ve P'nin farklı gruplarına giren noktalar ise

a, b, c, d sırasıyla SS, SD, DS, DD'nin sayılarını gösterecek olursa, $a+b+c+d=M$ veri seti içindeki tüm çiftlerin maksimum sayısını verir. $M=(N(N-1)/2)$ burada N veri seti içindeki toplam nokta sayısıdır.

Buradan C ve P arasındaki benzerliğin derecesi ölçülebilir.

- **Rand İstatistiği:** $R=(a+d)/M$

- **Jaccard Coefficient:** $J=a/(a+b+c)$

Aşağıda değerleri 0 ile 1 arasında değer alan başka benzerlik ölçüleri bulunmaktadır.

Bunlar;

- **Folkes ve Mallows İndeksi:**

$$FM = a / \sqrt{m_1 m_2} = \sqrt{\frac{a}{(a+b)} \cdot \frac{a}{(a+c)}} \quad (2.17)$$

Yukarıda anlatılan üç göstergenin yüksek değerleri C ile P arasında büyük bir benzerlik olduğunu gösterir. Diğer göstergeler;

- **Huberts Γ istatistiği:**

$$\Gamma = (1/M) \sum_{i=1}^{N-1} \sum_{j=i+1}^N X(i, j) Y(i, j) \quad (2.18)$$

Bu indeksin büyük değerleri X ve Y arasında güçlü bir benzerlik olduğunu gösterir.

- **Normalize Edilmiş Γ İstatistiği**

$$\bar{\Gamma} = [(1/M) \sum_{i=1}^{N-1} \sum_{j=i+1}^N (X(i, j) - \mu_x)(Y(i, j) - \mu_y)] / \sigma_x \sigma_y \quad (2.19)$$

Burada X(i,j) ve Y(i,j) X ve Y matrisinin (i,j) elemanları $\mu_x, \mu_y, \sigma_x, \sigma_y$ sırasıyla X ve Y matrisinin ortalamaları ve varyanslarıdır. Bu indeks değerleri -1 ile 1 arasında değer alır.

Tüm bu istatistikler sağ-yönlü olasılık yoğunluk fonksiyonuna sahiptir.

➤ **Bölünme P ile Yakınlık Matrisinin Karşılaştırılması:** Bölünme P aşağıdaki gibi gösterilebilir.

$g : X \rightarrow \{1 \dots nc\}$ Matris Y: $Y(i,j) = \{1, \text{Eğer } g(x_i) \neq g(x_j) \text{ ve } 0 \text{ diğer durumlarda}\}$
 $i, j = 1 \dots N$ Buradan yakınlık ölçüsü ve Y matrisi kullanılarak Γ istatistiği hesaplanabilir.

2.6.3.2 Dahili kriter

Kümeleme doğrulamanın bu yaklaşımı kullanılarak, sadece veri setinin doğasında olan özellikler ve nicelikler kullanılarak bir algoritmanın kümeleme

sonuçları değerlendirilebilir. Küme yapısına bağlı olarak, kümeleme doğrulamanın iki çeşit tane dahili kriteri bulunmaktadır. Bunlar: a) Küme şemalarının hiyerarşisi. b) Tek Küme şeması (HALKIDI vd, 2001).

- **Küme şemalarının hiyerarşine göre doğrulama:** Cophenetic matris olarak adlandırılan P_c matrisi, hiyerarşik algoritma tarafından üretilen hiyerarşik diyagram ile temsil edilebilir. Cophenetic matrisin $P_c(i,j)$ elamanı, başlangıç zamanında aynı küme içerisinde bulunan x_i ve x_j vektörlerinin yakınlık düzeyini gösterir. P_c ve P (Yakınlık Matrisi) arasındaki benzerliğin derecesini ölçmek için bir istatistiksel indeks tanımlanabilir. Bu indeks Cophenetic Korelasyon Katsayısı olarak adlandırılır ve aşağıdaki gibi tanımlanır.

$$CPCC = \frac{(1/M) \sum_{i=1}^{N-1} \sum_{j=i+1}^N d_{ij} c_{ij} - \mu_p \mu_c}{\sqrt{[(1/M) \sum_{i=1}^{N-1} \sum_{j=i+1}^N d_{ij}^2 - \mu_p^2][(1/M) \sum_{i=1}^{N-1} \sum_{j=i+1}^N c_{ij}^2 - \mu_c^2]}} \quad (2.20)$$

$$-1 \leq CPCC \leq 1$$

$M = N(N-1)/2$ ve N veri setindeki noktaların sayısını, μ_p, μ_c sırasıyla P ve P_c matrislerinin ortalamalarını göstermektedir.

$$\mu_p = (1/M) \sum_{i=1}^{N-1} \sum_{j=i+1}^N P(i, j), \quad \mu_c = (1/M) \sum_{i=1}^{N-1} \sum_{j=i+1}^N P_c(i, j) \quad (2.21)$$

Ayrıca d_{ij} , c_{ij} sırasıyla P ve P_c matrislerinin (i,j) elemanlarıdır. 0'a yakın indeks değeri iki matris arasındaki yüksek benzerliğin göstergesidir.

- **Tek Kümeleme Şeması Doğrulama:** Buradaki amaç, n_c tane küme içeren C kümeleme şeması ile yakınlık matrisi P 'nin arasındaki uyuşmanın derecesini bulmaktır. Bu yaklaşım için tanımlanan indeks Hubert Γ istatistiğidir.

2.6.3.3 Görelî kriter

Yukarıda anlatılan doğrulama metotları istatistiksel testlere dayanmaktadır. dahili ve harici kriterlere dayanan tekniklerin en büyük dezavantajı, yüksek hesaplama zamanı talep etmeleridir (HALKIDI vd, 2001). Görelî kriter yaklaşımı istatistiksel testler içermezler. Bu tekniğin temel fikri, önceden belirtilen kriterlere

göre tanımlanan şema kümeleri içerisinde en iyi kümeleme şemasını seçmektir. Bu problem aşağıdaki gibi tanımlanabilir: P_{alg} belirtilen kümeleme algoritmasının parametrelerinin kümesi olsun. P_{alg} 'nin farklı değerleri için C_i kümeleme şemaları arasından veri setine en uygun olan seçilir.

I) P_{alg} parametre olarak küme sayısını içermiyorsa: Bu durumda optimal parametrelerin seçimi şu şekilde yapılır: Algoritma, parametre değerlerinin geniş bir aralığı için çalıştırılır ve nc 'nin sabit kaldığı en geniş aralık seçilir. Bundan sonra, bu aralığın ortasına denk gelen değerler, P_{alg} parametrelerinin uygun değerleri seçilir. Aynı zamanda, bu prosedür veri seti için küme sayısını da belirler.

II) P_{alg} parametre olarak nc 'i içeriyorsa: En iyi kümeleme şemasını belirleyen prosedür doğruluk indeksine dayanır. En iyi performansı veren indeks q seçilir. Bunun için aşağıdaki adımlar takip edilebilir.

- Kümeleme algoritması nc 'nin n_{min} ve n_{max} arasında kalan bütün değerleri için çalıştırılır. Maksimum ve minimum değerler kullanıcı tarafından seçilir.
- nc 'nin her bir değeri için, algoritma, diğer parametrelerin farklı değerleri için r defa çalıştırılır (farklı başlangıç koşullarında).
- Elde edilen her bir nc değeri için q indeksinin en iyi değeri seçilir.

Bu adımlara dayanarak en iyi kümeleme şeması belirlenebilir. nc 'ye göre q indeksinin davranışına bağlı olarak en iyi kümeleme şemasını tanımlamak için iki yaklaşım sunulabilir. Doğruluk indeksi nc 'nin artan sayısı için artan ya da azalan bir trend izlemiyorsa, plot'un maksimum noktası bulunmaya çalışılır. Diğer taraftan, nc 'nin artan değerleri için artan ya da azalan bir trend varsa, bu indeks değeri için anlamlı bir değişimin olduğu değer aranır. Bu değişiklik plot'ta diz yeri gibi görünür ve ilgilenilen veri setinin küme sayısını gösterir.

2.6.3.3.1 Klasik kümelemede doğrulama kriteri

Klasik kümeleme algoritmalarına ilişkin birkaç tane kümeleme doğrulama metodu bulunmaktadır.

Modifiye edilmiş Hubert Γ istatistiği: modifiye edilmiş Hubert Γ istatistiği, aşağıdaki eşitlik tarafından tanımlanır.

$$\Gamma = (1/M) \sum_{i=1}^{N-1} \sum_{j=i+1}^N P(i, j) \Omega(i, j) \quad (2.22)$$

Burada $M=N(N-1)/2$, P veri setinin yakınlık matrisi, Ω x_i, x_j elemanlarının girdiği kümelerin temsili noktalarının (v_{ci}, v_{cj}) arasındaki uzaklığa eşit (i,j) elemanlarının $N \times N$ matrisidir.

Benzer olarak, eğer $d(v_{ci}, v_{cj}) d(x_i, x_j)$ 'ye yakınsa normalize edilmiş Hubert Γ istatistiği tanımlanabilir. P ve Ω aralarındaki uyuşma yüksekse, Γ ve normalize edilmiş Γ değerleri de büyük olacaktır. Karşıt şekilde, Γ 'nin yüksek değerleri yoğun kümelerin varlığını gösterir. n_c 'ye karşı Γ istatistiğinin grafiği çizilecek olursa, normalize edilmiş Γ 'daki önemli bir artma diz şeklinde görünen bir yapının oluşmasını sağlayacaktır. Bu meydana gelen şekil verilen veri seti için optimal küme sayısını verecektir.

Dunn ve Dunn'a Benzer Göstergeler: Klasik kümeleme için küme doğrulama kriterinden biri de yoğun ve iyi ayrılmış kümeleri belirlemek için Dunn tarafından önerilmiştir. Bu eşitlik aşağıdaki gibi tanımlanır:

$$D_{nc} = \min_{i=1, \dots, n_c} \left\{ \min_{j=i+1, \dots, n_c} \left(\frac{d(c_i, c_j)}{\max_{k=1, \dots, n_c} \text{diam}(c_k)} \right) \right\} \quad (2.23)$$

Burada $d(c_i, c_j)$ c_i, c_j kümeleri arasındaki benzemezlik fonksiyonu

$$d(c_i, c_j) = \min_{x \in c_i, y \in c_j} d(x, y) \quad (2.24)$$

ve $\text{diam}(c)$ ise kümelerin dağılım ölçüsü olarak düşünülebilen kümelerin çapıdır

$$\text{diam}(C) = \max_{x, y \in C} d(x, y) \quad (2.25)$$

Veri seti yoğun ve iyi ayrılmış kümeler içeriyorsa, kümeler arasındaki uzaklığı büyük, kümelerin çapının ise küçük olması beklenir. D_{nc} küme sayısına göre herhangi bir trend göstermez. Böylece, küme sayısına karşılık D_{nc} değerlerinin grafiğinin maksimumu, veri seti için uygun küme sayını verecektir.

Pal ve Biswas (1997) tarafından 3 tane gösterge önerilmiştir. Bu göstergeler daha çok aykırı değerlerin varlığı durumunda kullanılır. Bunlar Dunn indeksine dayandığı için Dunn benzeri indeksler olarak bilinir. Bundan başka, bu üç gösterge,

minimum karışılama ağacı (MKA) (minimum spanning tree) olarak bilinen kavramı kullanır. Bunun için kullanılan eşitlik aşağıdaki gibidir.

$$D_{nc} = \min_{i=1, \dots, nc} \left\{ \min_{j=i+1, \dots, nc} \left(\frac{d(c_i, c_j)}{\max_{k=1, \dots, nc} diam_k^{MKA}} \right) \right\} \quad (2.26)$$

Davies-Boldin (DB) indeksi: R_{ij} i. ve j. kümeler arasındaki benzerlik ölçüsü olduğu düşünülürse;

- 1) $R_{ij} > 0$
- 2) $R_{ij} = R_{ji}$
- 3) $s_i = 0$ ve $s_j = 0$ ise $R_{ij} = 0$
- 4) $s_j > s_k$ ve $d_{ij} = d_{ik}$ ise $R_{ij} > R_{ik}$
- 5) $s_j = s_k$ ve $d_{ij} < d_{ik}$ ise $R_{ij} < R_{ik}$

Bu koşullar altında R_{ij} simetriktir ve negatif değildir. Yukarıdaki koşulları sağlayan R_{ij} için basit bir seçim aşağıdaki gibidir.

$$R_{ij} = (s_i + s_j) / d_{ij} \quad (2.27)$$

ve DB indeksi aşağıdaki gibi tanımlanır.

$$DB_{nc} = \frac{1}{nc} \sum_{i=1}^{nc} R_i \quad (2.28)$$

Burada $R_i = \max_{i=1, \dots, nc} R_{ij}$ $i=1, \dots, nc$

Yukarıdaki eşitlikten açıkça görülmektedir ki; DB_{nc} her bir küme arasındaki ortalama benzerliktir. İstenilen durum kümelerin diğerlerine olan benzerliğinin minimum olmasıdır. Bu yüzden DB'yi minimum yapan kümeleme aranır. DB küme sayısına göre bir trend göstermez.

Klasik kümeleme için diğer doğruluk göstergeleri, Dave (1996) ve Milligan vd. (1983) vs. tarafından önerilmiştir. Ancak bu göstergelerin çoğu hesaplama açısından oldukça maliyetlidir.

KOKSS, YKR, RK, KU: Bunlar dışında, veri setindeki küme sayısını belirlemek için eş zamanlı olarak kullanılan 4 farklı doğruluk indeksi daha vardır. Bu indeksler

hiyerarşik kümeleme adımının her bir adımına uygulanır. Bu indekslere ilişkin tanımlar aşağıdaki gibidir:

KOKSS: Yeni kümenin Kök-Ortalama-Kare-Standart-Sapması: Kümeleme hiyerarşisinin bir düzeyinde tanımlanan yeni kümeleme şemasının KOKSS'i tüm değişkenlerin paylaştırılmış örnek varyansının kareköküdür. Bu indeks hiyerarşik algoritmanın her bir adımında şekillenen kümelerin homojenliğini ölçer. Kümeleme analizinin amacı homojen grupları ortaya çıkarmak olduğundan dolayı KOKSS mümkün olduğunca küçük olmalıdır. KOKSS değerlerinin, geçerli adımında, bir önceki adımından daha büyük olması halinde, bu yeni kümeleme şemasının homojen olmadığı anlamına gelmektedir.

$$\text{KOKSS} = \sqrt{\frac{\sum_{i=1}^{n1} (X_i - \bar{X})^2_{\text{örnek1}} + \sum_{j=1}^{n2} (X_j - \bar{X})^2_{\text{örnek2}} + \dots + \sum_{k=1}^{nB} (X_k - \bar{X})^2_{\text{örnekB}}}{(n1-1) + (n2-1) + \dots + (nB-1)}} \quad (2.29)$$

KT sembolüyle gösterilen ve kareler toplamı anlamına gelen tanım, $KT = \sum_{i=1}^n (X_i - \bar{X})^2$ şeklinde gösterilir. Buna ek olarak bazı ek semboller de vardır.

- i) **GKT:** Grup içi kareler toplamı
- ii) **GAKT:** Gruplar arası kareler toplamı
- iii) **TK:** Toplam kareler toplamı

YKRK (Yarı-Kısmi R Kare): Bu indeks, tek bir algoritma adımında kümeler birleştirildikten sonra, **homojenliğin kaybını** ölçer. Bu indeks değeri eğer 0 ise, iki kümenin birleştirilmesi ile elde edilen küme homojen demektir. Eğer bu indeksin değeri çok büyük ise o zaman elde edilen yeni küme heterojen demektir. YKRK elde edilen yeni kümeni GKT'si ile bu kümenin oluşturulmasını sağlayan iki kümenin GKT toplamları arasındaki farktır. Şöyle ki;

$$\text{YKS} = \text{GKT}_{\text{yeni küme(i. ve j. kümenin birleştirilmesi)}} - (\text{GKT}_i + \text{GKT}_j) \quad (2.30)$$

RK (R kare): GAKT nin TK'ya oranıdır. Bu indeks gruplar arasındaki farklılığı ölçer. $TK = GKT + GAKT$ olduğundan, ya GKT daha büyük olacaktır ya da GAKT büyük olacaktır. Bunun sonucunda gruplar arasındaki fark ne kadar büyükse her bir grup o kadar homojen olacaktır. RK, kümeler arasındaki farklılığın bir derecesi

olarak düşünülebilir. Bundan başka bu indeks gruplar arasındaki homojenliğin bir derecesini gösterir. Bu indeks 0 ile 1 arasında değer alır. Eğer bu değer 0 ise, kümeler arasında fark yoktur. Diğer taraftan eğer bu değer 1'e eşit ise bu, kümeler arasındaki önemli farklılığın göstergesidir.

KU (iki küme arasındaki uzaklık): Bu indeks herhangi bir adımda birleştirilen iki küme arasındaki uzaklığı ölçer. Bu indeksin değeri seçilecek uzaklık fonksiyonuna bağlıdır. Bu 4 indeks kullanılarak verilen veri seti için, en uygun küme sayısı belirlenebilir.

2.6.3.3.2 Bulanık kümelemede doğrulama kriteri

Bulanık doğrulama kriterleri temel olarak, bulanık bölünmenin yoğunluğunu ve ayrılmasını ölçmek için tasarlanır. Genelde, yoğunluk ortalama olarak her bir küme merkezinden piksel olarak sapmalar tahmin edilerek bulunur. Bölünmenin ayrılması ise kümeler arasındaki uzaklık ile temsil edilir. Bulanık kümeleme için kullanılan doğrulama indeksleri, sadece verilerin kümelere aitlik derecelerini gösteren üyelikler değerleri kullanılarak hesaplanan ve hem üyelik dereceleri hem veri seti kullanılarak hesaplanan doğruluk indeksleri olarak ikiye ayrılır. Sadece üyelik değerlerini kullanan doğrulama indeksleri olarak PC ve Bölünme Entropi Katsayısı verilebilir. Hem üyelik değerlerini hem veri setini kullanan doğrulama kriterleri arasında Xie-Beni, Fukuyama-Sugeno, Gath-Geva, Ortalama Bölünme Yoğunluğu, PD ve V_{sv} doğruluk indeksleri yer alır. Tüm doğrulama indeksleri, küme içi hatayı minimum, kümeler arası hatayı maksimum yapan küme sayısını bulmaya çalışır.

3. BULANIK MANTIK

3.1 Bulanık Mantığa Giriş

Bulanık mantık (Fuzzy Logic) kavramı ilk kez 1965 yılında California Berkeley Üniversitesinden Prof. A. Lotfi Zadeh'in bu konu üzerinde ilk makalelerini yayınlamasıyla duyuldu. O tarihten sonra önemi gittikçe artarak günümüze kadar gelen bulanık mantık, belirsizliklerin anlatımı ve belirsizliklerle çalışılabilmesi için kurulmuş katı bir matematik düzen olarak tanımlanabilir. Bilindiği gibi istatistikte ve olasılık kuramında, belirsizliklerle değil kesinliklerle çalışılır ama insanın yaşadığı ortam daha çok belirsizliklerle doludur. Bu yüzden insanoğlunun sonuç çıkarabilme yeteneğini anlayabilmek için belirsizliklerle çalışmak gereklidir.

Bulanık mantık ile matematik arasındaki temel fark bilinen anlamda matematiğin sadece aşırı uç değerlerine izin vermesidir. Klasik matematiksel yöntemlerle karmaşık sistemleri modellemek ve kontrol etmek işte bu yüzden zordur, çünkü veriler tam olmalıdır. Bulanık mantık kişiyi bu zorunluluktan kurtarır ve daha niteliksel bir tanımlama olanağı sağlar. Bir kişi için 38,5 yaşında demektense sadece orta yaşlı demek birçok uygulama için yeterli bir veridir. Böylece azımsanamayacak ölçüde bir bilgi indirgenmesi söz konusu olacak ve matematiksel bir tanımlama yerine daha kolay anlaşılabilen niteliksel bir tanımlama yapılabilecektir.

Bulanık mantıkta bulanık kümeler kadar önemli bir diğer kavramda sözel değişken (Linguistic Variable) kavramıdır. Sözel değişken "sıcak" veya "soğuk" gibi kelimeler ve ifadelerle tanımlanabilen değişkenlerdir. Bir sözel değişkenin değerleri bulanık kümeler ile ifade edilir. Örneğin oda sıcaklığı sözel değişken için "sıcak", "soğuk" ve "çok sıcak" ifadelerini alabilir. Bu üç ifadenin her biri ayrı ayrı bulanık kümeleri ile modellenir.

Bulanık mantığın uygulama alanları çok geniştir. Sağladığı en büyük fayda ise "insana özgü tecrübe ile öğrenme" olayının kolayca modellenebilmesi ve belirsiz kavramların bile matematiksel olarak ifade edilebilmesine olanak tanınmasıdır. Bu nedenle lineer olmayan sistemlere yaklaşım yapabilmek için özellikle uygundur.

Bulanık mantık konusunda yapılan araştırmalar Japonya'da oldukça fazladır. Özellikle bulanık süreç kontrolü olarak isimlendirilen özel amaçlı bulanık mantık mikroişlemci çipi'nin üretilmesine çalışılmaktadır. Bu teknoloji fotoğraf makineleri,

çamaşır makineleri, klimalar ve otomatik iletim hatları gibi uygulamalarda kullanılmaktadır. Bundan başka uzay arařtırmaları ve havacılık endüstrisinde de kullanılmaktadır. Tusaş Havacılık ve Uzay Sanayi (TAI)'de arařtırma geliřme kısmında bulanık mantık konusunda çalıřmalar yapılmaktadır. Yine bir bařka uygulama olarak otomatik civatalamaların deęerlendirilmesinde bulanık mantık kullanılmaktadır. Bulanık mantık yardımıyla civatalama kalitesi belirlenmekte, civatalama teknięi alanında bilgili olmayan kiřiler açasından konu Őeffaf hale getirilmektedir. Burada bir uzmanın deęerlendirme sınırlarına eriřilmekte ve hatta geçilmektedir.

Bulanık kuramının merkez kavramı bulanık kümelerdir. Küme kavramı kulaęa biraz matematiksel gelebilir ama anlařılması kolaydır. Örneęin “orta yař” kavramını inceleyerek olursak, bu kavramın sınırlarının kiřiden kiřiye deęiřiklik gösterdięini görürüz. Kesin sınırlar söz konusu olmadıęı için kavramı matematiksel olarak da kolayca formüle edemeyiz. Ama genel olarak 35 ile 55 yařları orta yařlılık sınırları olarak düşünülebilir. Bir bulanık kümesi kendi aitlik fonksiyonu ile aalık olarak temsil edilebilir. Aitlik fonksiyonu 0 ile 1 arasındaki her deęeri alabilir. Böyle bir aitlik fonksiyonu ile “kesinlikle ait” veya “kesinlikle ait deęil” arasında istenilen incelikte ayarlama yapmak mümkündür.

Bulanık mantık, İngilizcesiyle fuzzy logic, adından anlařılabileceęi gibi mantık kurallarının esnek ve bulanık bir Őekilde uygulanmasıdır. Klasik mantıkta bildięiniz gibi, "doęru" ve "yanlıř" yada "1" ve "0"lar vardır, oysa bulanık mantıkta, ikisinin arasında bir yerde olan önermeler ve ifadelere izin verilebilir ki, gerçek hayata baktıęımızda hemen hemen hiçbir Őey kesinlikle doęru veya kesinlikle yanlıř deęildir. Gerçek hayatta önermeler genelde kısmen doęru veya belli bir olasılıkla doęru Őeklinde deęerlendirilir. Bulanık mantıęa da zaten klasik mantıęın gerçek dünya problemleri için yeterli olmadıęı durumlar dolayısıyla ihtiyaç duyulmuřtur. Bulanık mantıęın sistemi řu Őekildedir. Bir ifade tamamen yanlıř ise klasik mantıkta olduęu gibi 0 deęerindedir, yok eęer tamamen doęru ise 1 deęerindedir (Ancak bulanık mantık uygulamalarının çoęu bir ifadenin 0 veya 1 deęerini almasına izin vermezler, veya sadece çok özel durumlarda izin verirler). Bunların dıřında tüm ifadeler 0'dan büyük 1'den küçük reel deęerler alırlar. Yani deęeri 0.32 olan bir ifadenin anlamı %32 doęru, %68 yanlıř demektir (URL4).

3.2 Bulanık Kümeler

Klasik küme teorisinde, bir eleman bir kümeye kesin olarak ya girer ya da girmez. İkisinin ortasında bir durumdan söz edilemez. Bulanık küme teorisi ise, elemanların farklı üyelik dereceleriyle birden fazla kümeye girmesini sağlayan, klasik küme teorisinin genişletilmiş versiyonudur. Üyelik fonksiyonları bir elemanın bir kümeye ne kadar ait olduğunu gösteren değerlerdir. 0 olması durumu, elemanın kümeye ait olmadığını, 1 olması durumu ise kesin olarak ait olduğunu gösterir. 1'ye yakın değerler elemanın yüksek derecede kümeye ait olduğunu, 0'a yakın değerlerde ise düşük derecede ait olduğunu gösterir.

Bu şekilde tanımlanan üyelik derecelerinin her bir bulanık söz için üç temel özelliği sağlaması tanım olarak gerekmektedir. Bunlar şöyle sıralanabilir:

- 1) Bulanık kümenin normal olmasıdır ki; bunun için en azından o küme de bulunan elemanlardan bir tanesinin en büyük üyelik derecesi olan 1'e sahip bulunması gerekliliğidir.
- 2) Bulanık kümenin monoton olması gerekir ki bunun anlamı üyelik derecesi 1'e eşit olan elemana yakın sağda ve soldaki elemanların üyelik derecelerinin de 1'e yakın olmasıdır.
- 3) Üyelik derecesi 1'e eşit olan elemandan sağa ve sola eşit mesafede hareket edildiği zaman bulunan elemanların üyelik derecelerinin birbirine eşit olmasıdır ki buna da bulanık kümenin simetrik özelliği adı verilir.

Klasik kümelerle bulanık kümelerin arasındaki önemli farklardan bir tanesi, klasik kümelerin sadece bir tane dikdörtgen üyelik derecesi fonksiyonu bulunmasına karşılık, bulanık kümenin yukarıdaki üç şarttan ilk ikisini mutlaka sağlayacak biçimde değişik üyelik derecesi fonksiyonlarına sahip olmasıdır. Buradan üyelik derecesi fonksiyonlarının mutlaka simetrik olma özelliğini sağlaması gerekmediği anlaşılabilir (Şen, 2001).

3.2.1 Bulanık kümelerin gösterimi

Verilen bir U evrensel kümesi için karakteristik fonksiyon f_A bulanık olmayan bir A kümesi için aşağıdaki şekilde tanımlanır.

$$f_A : U \rightarrow \{0,1\}$$

$$f_A(x) = 1 \quad x \in A \quad (3.1)$$

$$f_A(x) = 0 \quad x \notin A$$

Bulanık kümeler ise $[0,1]$ aralığında değişen üyelik derecelerini kullanır.

$$\mu_A : U \rightarrow [0,1]$$

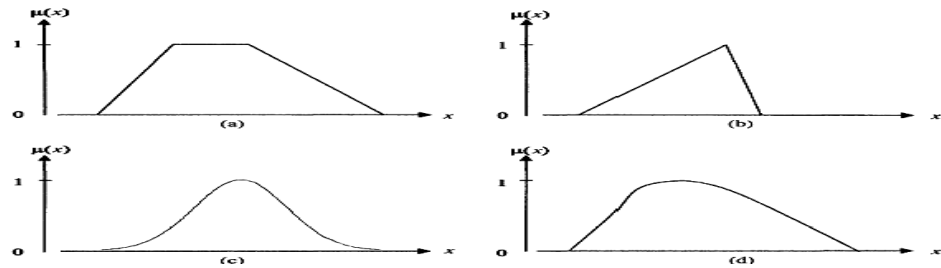
Bulanık kümeler 4'e yakın sayılar, siyaha yakın renkler, büyük gezegenler, yüksek sıcaklık, kısa uzaklık gibi kesin olmayan kavramları temsil eder. Bulanık küme alanları kesikli veya sürekli olabilir. Sonlu alana sahip bulanık küme A için yaygın olarak matematiksel notasyon $\{x_1, x_2, \dots, x_n\}$ ve üyelik fonksiyonu μ_A ise:

$$\mu_A(x_1)/x_1 + \mu_A(x_2)/x_2 + \dots + \mu_A(x_n)/x_n \quad (3.2)$$

Örnek olarak 4'e yakın sayılar için üyelik fonksiyonu

$$0.2/1 + 0.5/2 + 0.8/3 + 1/4 + 0.8/5 + 0.5/6 + 0.2/7$$

Çoğu durumunda, bulanık kümeler sürekli alanlardan oluşur. Bu durumda üyelik fonksiyonları için matematiksel fonksiyonlar ve grafikler tanımlanır. Çoğu durumda, üçgen, yamuk, s-şekilli ve gauss fonksiyonları tercih edilir.



Şekil 3.1 Yaygın olarak kullanılan üyelik fonksiyonlar. (a) Yamuk Şekilli. (b) Üçgen Şekilli. (c) Gauss Üyelik Fonksiyonu. (d) Çan Şekilli. Not: Tüm üyelik fonksiyonları , sürekli, normal ve konvekstir.

Üçgen şekilli bulanık küme aşağıdaki gibi tanımlanabilir.

$$\begin{aligned}\mu_A(x) &= (x-a)/(b-a) & x \leq b \\ \mu_A(x) &= (c-x)/(c-b) & x > b\end{aligned}\quad (3.3)$$

Burada a üyelik fonksiyonu alt sınırını, b üyelik değeri 1'e eşit olan elemanı (tepe noktası), c ise üyelik fonksiyonu üst sınırını göstermektedir.

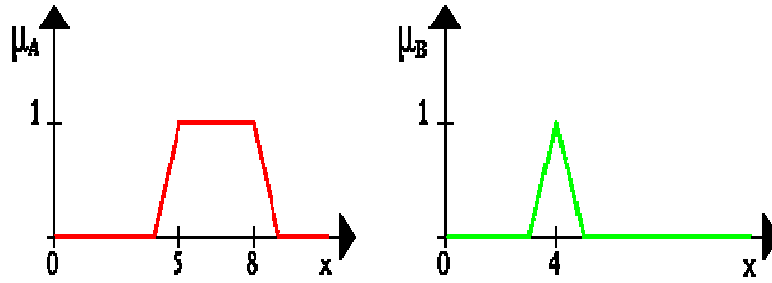
Üçgen şekilli üyelik fonksiyonu düşünüldüğünde; üyelik değeri 1 olan elemanın olduğu kısım fonksiyonun **özü**, üyelik derecesi 0 ile 1 arasında değişen elemanların olduğu kısım fonksiyonun **kısımları** üyelik fonksiyonu 0 den büyük olan elemanların oluşturduğu kısım ise fonksiyonunun **dayanağı** olarak adlandırılır. Normal olma özelliğinden dolayı üyelik fonksiyonunun değeri en az bir eleman için 1 olmalıdır. Ayrıca üyelik fonksiyonları sürekli artan ve sürekli azalan özelliğine sahip olmalıdır. Üyelik değeri 0.5 olan kısım **geçiş noktası**, en yüksek üyeliğe sahip olan kısım ise **yükseklik** olarak adlandırılır.

Bulanık küme kavramı için önemli olan bir diğer kavram ise sözel değişken kavramıdır. Cebir değişkenleri sayısal değerler alırken, sözel değişken kelime veya cümle şeklindeki metin değerler almaktadır. Bu değerlerin kümesi terim kümesi olarak adlandırılır. Terim kümesi içindeki her bir değer, temel değişkenlere dayanarak tanımlanmış bulanık bir değişkendir. Kısaca hiyerarşik olarak bakıldığında sözel değişken → bulanık değişken → temel değişken

Sözel değişkenin “yaş” olarak etiketlendiği varsayılırsa; bu sözel değişkenin her biri bir bulanık kümeye karşılık gelen, terimleri, “yaşlı”, “çok yaşlı”, “orta yaşlı”, “biraz genç”, “genç”, “çok genç” şeklinde olabilir. Her bir terim, 0 ile 100 arasında ölçeklendirilebilen temel değişkenlerin üzerinde tanımlanan bulanık bir değişkendir (Geraldo).

3.3 Bulanık Küme İşlemleri

Klasik kümeler üzerinde yapılan işlemler, bulanık kümeler kullanılarak yapılabilir. Yaygın olarak kullanılan bulanık kümeleme işlemleri birleşim, kesişim, tümleyendir. Klasik küme işlemlerinden farkı, 0 ile 1 aralığındaki bütün değerleri dikkate almasıdır.

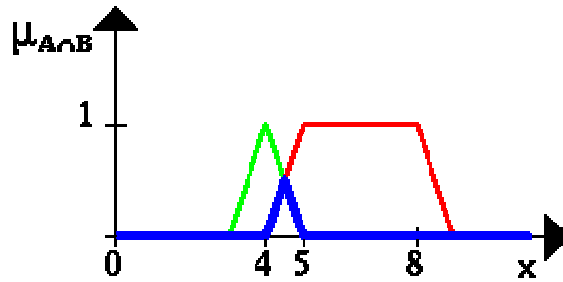


Şekil 3.2 A ve B bulanık kümelerine ilişkin üyelik fonksiyonları

3.3.1 Bulanık kesişim

İki bulanık kümenin kesişimi, \wedge ile temsil edilir. Klasik küme işlemlerindeki, kesişim işlemi iki kümenin ortak elemanlarının oluşturduğu kümeyi bulmak için kullanılır. Bulanık kümelerde kesişim kümesindeki elemanların üyelik derecelerinin bulunması için, tüm bulanık kümelerdeki elemanların üyelik derecelerinin minimumu hesaplanır. Şöyle ki; A_1, A_2, \dots, A_n bulanık kümeler olsun

$$\mu_{A(x) \wedge A_2(x) \wedge \dots \wedge A_n(x)} = \min\{\mu_{A_1}(x), \mu_{A_2}(x), \dots, \mu_{A_n}(x)\} \quad (3.4)$$



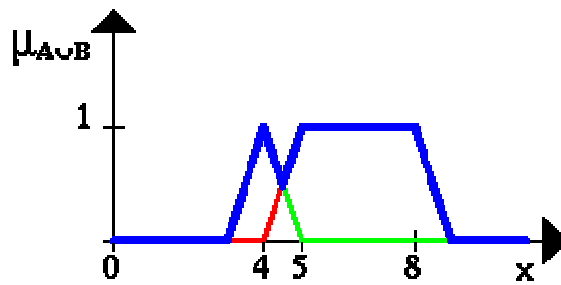
Şekil 3.3 A ve B bulanık kümelerinin kesişimi.

Şekil 3.3'den görüldüğü gibi A ve B bulanık kümelerinin kesişimin ilişkin üyelik fonksiyon grafiği gösterilmiştir.

3.3.2 Bulanık birleşim

İkili mantıkta VEYA işlemine karşılık gelmektedir ve \vee işareti ile gösterilir. İki bulanık kümeye ilişkin birleşim kümesi bulunurken, klasik mantıkta olduğu gibi her iki kümedeki elemanların (tekrar edenler hariç) her biri alınır. Bu birleşim kümesine ilişkin üyelik fonksiyonları ise her bir kümeye ait elemanların üyelik değerlerinin en büyüğü alınır. A_1, A_2, \dots, A_n bulanık kümeleri için birleşim işlemi:

$$\mu_{A(x) \vee A_2(x) \vee \dots \vee A_n(x)} = \max\{\mu_{A_1}(x), \mu_{A_2}(x), \dots, \mu_{A_n}(x)\} \quad (3.5)$$



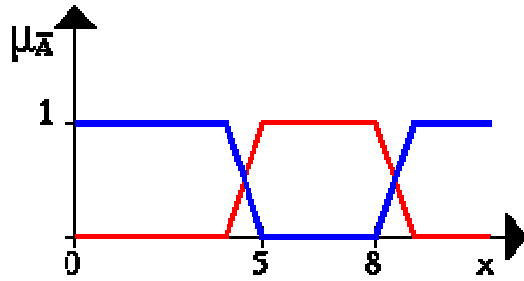
Şekil 3.4 A ve B bulanık kümelerine ilişkin birleşim işlemi.

Şekil 3.4'ten A ve B bulanık kümelerinin birleşiminin üyelik fonksiyonu gösterilmiştir.

3.3.3 Bulanık tümleyen

Bir bulanık kümenin tümleyeni, bu kümeye ilişkin elemanların dışında bütün elemanları içeren küme olarak tanımlanır ve A bulanık kümesi için, tümleyeni \bar{A} şeklinde gösterilir. Tümleyen bulanık kümeyle ilişkin üyelik fonksiyonu ise aşağıdaki gibi tanımlanır:

$$\mu_{\bar{A}} = (1 - \mu_A) \quad (3.6)$$



Şekil 3.5 A bulanık kümesi için tümleyen işlemi.

3.3.4 Kartezyen çarpım

Genel bulanık ilişkiler; aynı zamanda bulanık kümeler olduğu için, kartezyen çarpımı iki ya da daha fazla bulanık kümeler arasındaki ilişki olarak tanımlayabiliriz. A, X evrensel kümesinde B ise Y evrensel kümesinde bulanık kümeler olsun. A ve B'nin kartezyen çarpımı bulanık ilişki R'yi verecektir.

$$A \times B = R \subset X \times Y$$

Bulanık ilişki R ise aşağıdaki üyelik fonksiyonuna sahiptir.

$$\mu_R = \mu(x, y) = \min(\mu_A(x), \mu_B(x)) \quad (3.7)$$

$A \times B = R$ olarak tanımlanan kartezyen çarpım iki vektörün çapraz çarpımları gibidir. $r = 2$ durumunda kartezyen çarpım kümeler arası elemanların eşleştirilmesi düşüncesine dayanır. Bulanık kümelerin her biri, üyelik değerleri vektörü olarak düşünülebilir (Şahinli, 1999).

3.3.5 Diğer bulanık küme işlemleri

A, B, C, X evreninde tanımlı üç bulanık küme olmak üzere bunlar arasındaki tanımlanabilecek diğer küme işlemleri şu şekildedir:

Eşitlik : A ve B bulanık kümeleri eğer onların tüm elemanları için üyelik değerleri eşitse eşit iki kümedir.

$$A \rightarrow \mu_A(x), \quad B \rightarrow \mu_B(x), \quad \forall x \in A, x \in A \quad \mu_A(x) = \mu_B(x), \quad X \in E \quad (3.8)$$

Daraltma: A bulanık kümesinin üyelik fonksiyon değerlerinin azaltılmasını hedefleyen bir işlem gerçekleştirilir. Değerlerin işlem sonucunda küçülmesi bu değerlerin 1'den küçük olmasından kaynaklanır. Bu işlem üyelik fonksiyonları 1'e yakın olan elemanlara ağırlık kazandırır.

$$\mu_{\text{con}}(A)(x) = (\mu_A(x))^2 \quad \forall x \in X \quad (3.9)$$

Genişletme: A bulanık kümesi üzerinde yayılma işleminin yapılmasıyla aşağıdaki ifade gerçekleşir.

$$\mu_{\text{DIL}}(A)(x) = \sqrt{\mu_A(x)} \quad \forall x \in X \quad (3.10)$$

Yoğunlaştırma: Yoğunlaştırma işlemiyle bulanık küme klasik kümeye yaklaştırılır. Nitekim işlem sonucunda üyelik fonksiyonunun değeri 0,5'den büyük olan elemanların üyelik fonksiyon değerleri artar, 0,5'den küçük olan elemanların üyelik dereceleri azalır.

$$\mu_{\text{nit}}(A)(x) = \begin{cases} 2(\mu_A(x))^2, & 0 \leq (\mu_A(x)) \leq 0.5 \\ 1-2(1-\mu_A(x))^2, & 0.5 \leq (\mu_A(x)) \leq 1 \end{cases} \quad (3.11)$$

$$\text{Toplam : } A \oplus B \leftrightarrow \mu_A(x) + \mu_B(x) - \mu_A(x)\mu_B(x) \quad (3.12)$$

$$\text{Çarpım: } AB \leftrightarrow C(x) = \mu_A(x)\mu_B(x) \quad (3.13)$$

$$\text{Sınırlı Toplam: } A \oplus B \leftrightarrow C(x) = (\mu_A(x) + \mu_B(x)) \wedge 1 \quad (3.14)$$

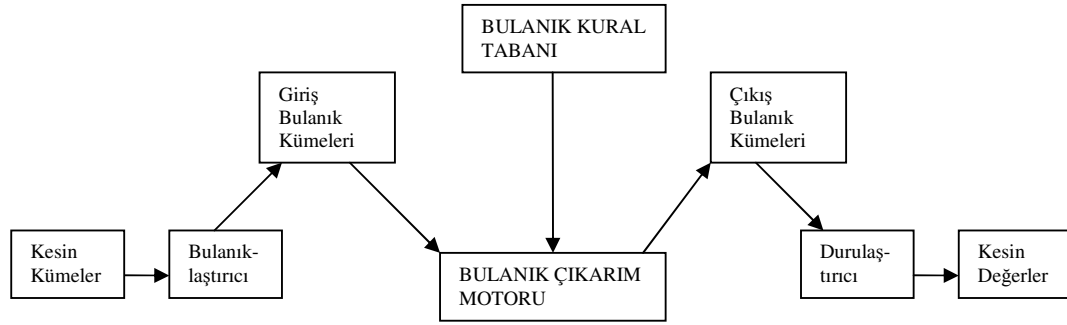
$$\text{Sınırlı Fark : } A - B \leftrightarrow (\mu_A(x) - \mu_B(x)) \vee 0 \quad (3.15)$$

(Şahinli, 1999).

3.4 Bulanık Sistemler

3.4.1 Bulanık sistemlerin yapısı

Bulanık sistem terimi, bulanık teoriye dayanan bazı mekanizmalara ve mimarilere sahip herhangi bir sistemi etiketlemek için kullanılabilir. Bulanık sistem genel olarak iki ana birimden oluşur. Bunlar sistemin kesin parçalarını bulanığa dönüştüren, bulanık parçalarını ise kesine dönüştüren birimlerdir. Kesinden bulanığa dönüştüren birim bulanıklaştırıcı, bulanıktan kesine dönüştüren birim ise durulaştırıcı olarak adlandırılır.



Şekil 3.6 Genel bulanık sistem.

Bir bulanık sistemin orijinal önerisi, Bulanık Çıkarım Mekanizmasının içindeki kurallar kümesi tarafından meydana gelen ana mekanizmayı kullanmaktır. Son zamanlarda, bulanık sistemler, regresyon modelleri, bulanık dilbilgisi, yapay sinir ağları, genetik algoritmaları, bulanık toplama fonksiyonlar, Petri-ağları, gibi teknikler içerir.

Bulanık sistemler, izleme, kontrol etme, tanıma, sorgulama ve optimize etme gibi farklı görevleri yerine getirmek için kullanılır. Açıkça; 4 kategoriye bölünebilir:

1. Bir problem için özel kararlar alınmasını gerektiren sıkı kuralları olan sistemler.
2. Problemin teşhisini sadece deneme yoluyla yapan tanımlayıcı sistemler, sınıflandırma, desen tanıma gibi.
3. Bazı performans kriterlerinin üstesinden gelmek için koşul ve etkileri saptamayı deneyen optimizasyon sistemleri.
4. Gelecekteki sonuçları tahmin etmeyi deneyen tahmin edici sistemler.

Bulanık sistemin geliştirilmesi için gerçekleştirilmesi gereken birkaç adım vardır:

1. Sistemin ölçülebilir özelliklerini tanımla. Bu özellikler kesin değerler olmamalıdır, çünkü bulanık sistemler belirsizlik ile ilgilenir.
2. Her bir özelliği sözel değişken olarak tanımla. Bu değişkenleri tanımlayan uygun sözel terimlerin kümesini oluştur. Burada üyelik fonksiyonları seçmeye gerek yoktur.
3. Bulanık sistemden istenilen sonuç elde edilene kadar 1. ve 2. adımı tekrarla

4. Sözel terimler için üyelik fonksiyonlarını oluştur.
5. Bulanık sistemin istenilen çıktısı ile sistemin özellikleri arasındaki ilişkileri tanımlayan bilgiyi ortaya çıkar.
6. Sistem düzgün çalışıyorsa ve sonuçların doğruluğu kontrol edildiyse işlemi tamamla
7. Eğer test aşaması başarısız sonuçlar verdiyse, sistemde değiştirilebilir yerleri analiz et ve geri dön
8. Sonuç sistemi onayla

Bu adımlar genel olarak bulanık sistemin çalışma stratejisini anlatmaktadır. Şu açıktır ki; bulanık sistemi tasarlamak deneme yanılma yoluyla olmaktadır. Bu arada önemli olan bir nokta da sistemin anlamsal yapısıdır (semantik). Örneğin, farklı uzman düşüncelerini birleştirmek için bazı toplama operatörleri kullanıldığında, bu operatörlerin nasıl kullanılacağını bilmemiz gerekir.

3.4.2 Bulanıklaştırma

Bütün bulanık sistemler için gerçekleştirilmesi gereken ilk adım, girdileri geleneksel kesin uzaydan, bulanık uzaya dönüştürmektir. Bu adım, bulanıklaştırma ya da bulanık dönüştürücü olarak bilinir ve girdi değerleri için olan belirsizliği teşhis eder. Her girdi kümesi, sözel bir değişken ile birleştirilir. Her bir sözel değişken için, değişkeni subjektif olarak tanımlayan sözel terimler kümesi oluşturulmalıdır. Çoğu zaman, sözel terimler “sıcak”, “geniş” gibi sözel değişkenin büyüklüğünü tanımlayan kelimelerdir. Her sözel terim bir bulanık kümeye ve onun sahip olduğu üyelik fonksiyonuna karşılık gelmektedir. Burada umulan şey sözel değişkenin ilgilenilen alanı tamamen kaplamasıdır. Bundan başka benzer kavramları tanımlayan sözel terimlerin kesişimin olması gerekir. Örnek olarak “sıcaklık” tan bahsedilecek olursa, “soğuk ” ve “çok soğuk” tanımlayan sözel terimlerin ortak bazı değerleri olmalıdır. Genellikle bitişik sözel terimler %10 ile %50 arasında üst düşüme sahiptir. Her bir sözel değişkene uygun olan, verilen tüm bulanık kümeler için bulanıklaştırmanın anlamı, bulanık kümelerdeki bütün girdi değerlerin üyelik değerlerini hesaplamaktır.

3.4.3 Üyelik fonksiyonlarının belirlenmesi

Bulanık kümeleri tanımlayan üyelik fonksiyonlarını karşılaştırmak, gerçek zamanlı, deneysel problemler üzerinde çalışma açısından çok önemlidir. Üçgen ve yamuk fonksiyonları birçok duruma uygun olduklarından ve hızlı hesaplama zamanına sahip olduklarından genellikle tercih edilen fonksiyonlardır. Diğer Gaussian ve sigmoid gibi eğrisel üyelik fonksiyonları ise düzgün (smooth) sonuçlar sağlar. Son olarak keyfi şekilli üyelik fonksiyonları kullanılabilir.

Üyelik fonksiyonları uzman bilgisinden yararlanılarak ve verilen veri setini kullanarak iki şekilde belirlenebilir. Uzman bilgilerinden yararlanılarak üyelik fonksiyonlarını belirlemek için bazı yöntemler bulunmaktadır. Bunlar:

1. **Sezgi:** Tasarımcının önceki bilgilerinden yararlanılarak hangi eğrinin kullanılacağına karar verilir.
2. **Yatay Metot** (Pedrycz vd, 1998): Elemanların üyelik değerleri hakkındaki bilgi, uzmanların ilgilenilen kavram ile uyumlu “evet”, “hayır ” şeklinde oy vermesi ile elde edilir. Tahmin edilen üyelik değeri pozitif cevapların toplam cevaplara oranıdır.
3. **Dikey Metot** (Pedrycz vd, 1998): Bu metotta küme α kesmesi kullanılarak oluşturulur. Buradaki soru şudur; “x elemanı α % kümeyi tanımlayan kavramla uygun mu ?” α -kesmeleri bulanık önerme olarak göstermek mümkündür.
4. **İkili-Karşılaştırma Metodu** (Saaty, 1980): Bazı kesikli öncelik seviyeleri kullanılarak, her bir çift karşılaştırıldığında, matematiksel operatörler üyelik değerlerini belirleyebilir.

Bulanık Kümeleme Teknikleri: Veri setini kullanarak üyelik fonksiyonlarının belirlenmesi için bulanık kümeleme teknikleri kullanılabilir. Bulanık kümeleme teknikleri üyelik fonksiyonlarını belirlemek için uygun bir araçtır ve bu işlemi aşağıdaki adımlarla ile gerçekleştirir:

- Önce girdi-çıktı verileri için kümeleme algoritması uygulanır. Sonra bölümlenme matrisinin girdi değişkenlerine projeksiyonu yapılarak tüm girdi değişkenleri için üyelik değerleri belirlenir.
- Bu girdi üyelik değerlerine göre yakınsama metodu uygulanarak onların ilgili üyelik fonksiyon şekilleri belirlenir (Türkşen,URL8).

3.4.4 Bulanık kurallar

En yaygın kullanılan bulanık sistemler bilgiyi temsil etmek için kuralları kullanırlar. Kurallar için tek bir tip olmamasına rağmen, IF-THEN şeklindeki kurallar standarttır. Basit bir IF-THEN kuralı aşağıdaki gibi yazılabilir.

$$\text{IF } a_1 \text{ and } a_2 \dots \text{ and } a_n \text{ THEN } b, \quad (3.16)$$

Burada a_i $1 \leq i \leq n$ ve b birer bulanık önermedir. Uzman sistemlere benzer olarak, her bir kural sistemin çıktısı ile ilgili uygun varsayımları temsil eder (Şen, 2001). Bulanık kurallar, kuralın sol ve sağ tarafındaki değişken sayılarına ve kuralın sonuç (consequent) kısmının biçimine göre aşağıdaki gibi iki ana başlık altında toplanır.

A) Girdi ve Çıktı Değişken Sayılarına Göre;

1) SISO (Single Input Single Output): Tek Girdili Tek Çıktılı bulanık kural aşağıdaki gibi yazılır: R tane kural olduğu düşünülürse

$$R^i : \text{IF } x \text{ is } A_i \text{ THEN } y_i \text{ is } B_i \quad i=1, \dots, c$$

Burada A_i 'ler girdi bulanık kümeler B_i 'ler ise çıktı bulanık kümelerdir.

2) MISO (Multi Input Single Output): Çok Girdili Tek Çıktılı bulanık kural aşağıdaki gibidir: R tane kural için;

$$R^i : \text{IF } x_1 \text{ is } A_{i1} \text{ and (or) } x_2 \text{ is } A_{i2} \text{ and (or) } \dots x_p \text{ is } A_{ip} \text{ THEN } y \text{ is } B_i \quad i=1, 2, \dots, c \quad j=1, 2, \dots, p$$

Burada $A_{i1}, A_{i2}, \dots, A_{ip}$ 'ler girdi bulanık kümeleri, B_i çıktı bulanık kümeyi göstermektedir.

3) MIMO (Multi Input Multi Output): Çok Girdili Çok Çıktılı bulanık kural aşağıdaki gibi yazılır: R tane kural için

$$R^i : \text{IF } x_1 \text{ is } A_{i1} \text{ and (or) } x_2 \text{ is } A_{i2} \text{ and (or) } \dots x_p \text{ is } A_{ip} \text{ THEN } y_1 \text{ is } B_{i1} \text{ ALSO } y_2 \text{ is } B_{i2} \text{ ALSO, } \dots, y_q \text{ is } B_{iq} \quad i=1, 2, \dots, c \quad j=1, 2, \dots, p \quad k=1, 2, \dots, q$$

Burada $A_{i1}, A_{i2}, \dots, A_{ip}$ girdi bulanık kümeleri $B_{i1}, B_{i2}, \dots, B_{iq}$ ise çıktı bulanık kümeleri göstermektedir.

B) Sonuç (consequent) kısmının biçimine göre;

1) Kuralın sonuç (consequent) kısmı bulanık kümelerden oluşuyorsa bulanık kural aşağıdaki gibi yazılır (Mamdani Kuralı):

IF x_1 is A_1 and x_2 is A_2, \dots, x_p is A_p THEN y is B

IF x_1 is A_1 and x_2 is A_2, \dots, x_p is A_p THEN y_1 is B_1 and y_2 is B_2, \dots, y_q is B_q

$j=1,2,\dots,p, k=1,2,\dots,q$

2) Kuralın sonuç kısmı her bir kural için lineer bir fonksiyondan oluşuyorsa bulanık kural ise şu şekildedir (Takagi-Sugeno Model);

IF x_1 is A_1 and x_2 is A_2, \dots, x_p is A_p THEN $y = f(x)$

Burada $f(x) = a_0 + a_1x_1 + \dots + a_qx_q$ şeklinde lineer bir fonksiyonu göstermektedir.

3) Kuralın sonuç kısmı sayısal bir değerden oluşuyorsa kural aşağıdaki gibi yazılır (Bulanıklık tekillik=Singleton).

IF x is A THEN y is b

3.4.5 Çıkarım mekanizması

Bulanık kural tabanında giriş çıkış bulanık kümeleri arasında kurulmuş olan ilişkilerin hepsini bir araya toplayarak sistemin bir çıkışlı davranmasını temin eden işlemler topluluğunu içeren mekanizmadır. Bu mekanizma, her bir kuralın çıkarımlarını bir araya toplayarak tüm sistemin girdiler altında nasıl bir çıktı vereceğinin belirlenmesine yarar (Şen, 2001). Basit bulanık sistemlerde, tüm kurallar paralel çalışır. Bir bulanık çıkarım mekanizmasının çalışma şekli iki adımda görülebilir:

- Her bir kural için sonuç (consequent) değerlerini hesaplamak.
- Birleştirilmiş bir sonuç hesaplamak.

Bu bulanık sistemlerde en basit ve en yaygın kullanılan düzensel çıkarım mekanizmasıdır. İlk adımı çalıştırmak için, sebep değerlerine dayanan sonuç değerlerini hesaplayan bir yöntem gereklidir. Bunun için bulanık ifadeleri temsil eden bir fonksiyona ihtiyaç vardır. Bu ifadelerden, tek bir kural için çıkarım mekanizması sağlayan genelleştirilmiş *modus ponens* kullanılarak anlamlı sonuçlar çıkartılabilir. Ancak ihtiyaç duyulan birleştirilmiş tek bir sonuçtur. Bunun için, tüm kuralları bir araya getirip tek bir sonuç üretilmesini sağlayan iki ana metot bulunmaktadır. Min-Max (Mamdani) ve bulanık katkı (additive) metodu. Mamdani yönteminde bulanıklaştırılan giriş bilgileri önceden oluşturulmuş kurallara tabi

tutulur. Kurallar uygulanırken, şartlar arasındaki [ve] durumunda giriş üyelik değerlerinden en küçük olanı, [veya] durumunda ise en büyük olanı alınarak çıkış üyelik fonksiyonunda bu sayının altında kalan alan bulunur. Örnek olarak aşağıdaki çıkarım işlemleri verilebilir.

Kural yapısının R^i : IF x_1 is A_{i1} and x_2 is A_{i2} and ... x_p is A_{ip} THEN y is B_i olması durumunda çıkarım işlemi şu şekilde yapılabilir:

$$\mu_B(y) = \max(\max\{\mu_{A_1}(x_1) \wedge \mu_{A_2}(x_2) \wedge \dots \wedge \mu_{A_p}(x_p)\} \wedge \mu_{B_i}(y)) \quad (3.17)$$

Kural yapısının IF x_1 is A_{i1} and,..... x_p is A_{ip} THEN $y^i = a_{i0} + a_{i1}x_1 + \dots + a_{ik}x_k$ şeklinde olması durumunda ise çıkarım işlemi aşağıdaki gibi yapılabilir:

$$y = \frac{\sum_{i=1}^c w_i(y^i)}{\sum_{i=1}^c w_i} \quad (3.18)$$

Burada w_i 'ler girdi üyelik fonksiyonlarının birleştirilmesinden elde edilmiş üyelik fonksiyonunu, y^i 'ler ise i . kuralın lineer fonksiyonunu göstermektedir.

3.4.6 Durulaştırma

Durulaştırma birimi, adından anlaşılacağı gibi bulanık bir niceliği, kesin bir niceliğe dönüştürür. Kontrol sistemlerinin hemen hemen hepsi bu geri dönüşümü gerektirir. Çıkarım işleminden sonra durulaştırma işlemi kuralın yapısına bağlı olarak yapılır. Eğer çıkarım sonucunda bulanık küme elde ediliyorsa durulaştırma işlemi yapılır. Aksi takdirde yapılmaz. Örnek olarak;

R^i : IF x_1 is A_{i1} and x_2 is A_{i2} and ... x_p is A_{ip} THEN y is B_i bulanık kurallarının sonucu bir bulanık kümeye karşılık geldiğinden çıkarım işleminden sonra durulaştırma işlemi yapılmalıdır. Bununla birlikte;

R^i : IF x_1 is A_{i1} and x_2 is A_{i2}, \dots, x_p is A_{ip} THEN $y = f(x)$ bulanık kurallarının sonucu ise sabit bir fonksiyona karşılık geldiğinden durulaştırma işleminin yapılmasına gerek yoktur. Durulaştırma işlemi gerektiği durumda bu işlemi gerçekleştirmek için birçok yöntem bulunmaktadır. Bunlardan bazıları şu şekildedir:

1) Maksimum-Üyelik Prensibi: Bu yöntemin diğer adı, **yükseklik** yöntemidir. Bu yöntemin uygulanabilmesi için, uç noktaları olan sonuç bulanık kümelerine ihtiyaç vardır. Bu durulaştırma işleminden elde edilen sonuç üyelik fonksiyonların en büyüğüdür ve aşağıdaki şekilde gösterilebilir.

$$\mu(z^*) \geq \mu(z) \quad \text{tüm } z \in Z \quad (3.19)$$

2) Maksimum Ortalama Prensibi: Bu yöntem sonuç bulanık kümelerinin en büyük üyeliklerinin ortalamasını verir. Bu bakımdan birinci yöntemle benzer yanları vardır.

3) En Büyük İlk veya Son Üyelik Derecesi Prensibi: Bu yöntemde, tüm çıktılardan birleşimi olarak ortaya çıkan bulanık kümede en büyük üyelik derecesine sahip olan en küçük (veya en büyük) bulanık küme değerini seçmek esasına dayanır.

4) Centroid (küme ağırlığı) Prensibi: Durulaştırma işleminde en yaygın kullanılan yöntemlerden biridir. Bu yöntemde sonuç bulanık kümelerinin ağırlık merkezi hesaplanır. Bu yönteme ilişkin notasyon şu şekildedir.

$$z^* = \frac{\int \mu(z)zdz}{\int \mu(z)dz} \quad (3.20)$$

5) Toplamların Merkezi Prensibi: Kullanılan durulaştırma işlemleri arasında en hızlı olanı bu yöntemdir. İki bulanık kümenin birleşimi yerine onların cebirsel toplamları kullanılır. Durulaştırılmış değer

$$z^* = \frac{\int z \sum_{k=1}^n \mu(z)dz}{\int \sum_{k=1}^n \mu(z)dz} \quad (3.21)$$

6) En Büyük Alan Merkezi Prensibi: Eğer sonuç bulanık kümesi, en azından iki tane dış bükey alt bulanık kümeyi içeriyor ise, dış bükey bulanık kümelerin en büyük alanlarının ağırlık merkezi durulaştırma işleminde kullanılır. Matematiksel hesaplanması ise;

$$z^* = \frac{\int \mu_{eb\zeta}(z)zdz}{\int \mu_{eb\zeta}(z)dz} \quad (3.22)$$

eşitliğine göre yapılır.

7) Ağırlıklı Ortalama Yöntemi: Bunun kullanılabilmesi için simetrik üyelik fonksiyonunun bulunması gereklidir. İşlemler matematik olarak (Şen, 2001),

$$z^* = \frac{\sum \mu_{\zeta}(\bar{z}) \cdot \bar{z}}{\sum \mu_{\zeta}(\bar{z})} \quad (3.23)$$

4. BULANIK KÜMELEME ALGORİTMALARI

4.1. Bulanık Kümeleme Yaklaşımları

Bulanık kümeleme analizi, desen tanıma, görüntü işleme ve bulanık modelleme gibi uygulamalar için etkili bir şekilde kullanılmaktadır. En iyi bilinen bulanık kümeleme yaklaşımı ilk olarak Dunn (1974) tarafından önerilen ve Bezdek tarafından geliştirilen Bulanık C-Ortalamlar (BCO) algoritmasıdır.

Bulanık kümeleme metotları ikiye ayrılabilir. Bir tanesi bulanık kümeleme için bulanık ilişkileri, diğeri ise amaç fonksiyonunu kullanır. Bulanık ilişkilere dayalı kümeleme adından anlaşılacağı gibi, orijinal bireyler arasındaki ilişkisel yapıyla ilgilenir. İlişki veya önermeler bireyler arasındaki benzerlik veya benzemezlik kavramlarına göre tanımlanabilir. Diğeri bir önerme ise bireyler arasındaki benzemezliği veya farkların derecesini ifade eden bir önerme gibi, tamamlayıcı karakterlerden biri olabilir.

Amaç fonksiyonuna dayalı algoritmalar ise kümeleme problemini optimizasyon problemi haline dönüştürür. Bu yöntemde küme içindeki benzersizliği ölçmek için amaç fonksiyonu kullanılır ve bu amaç fonksiyonu minimize edilerek en iyi bölünme elde edilir. Bu algoritmaların kullanılabilmesi için küme sayısının ve küme prototiplerinin nasıl olacağını bilmesi gerekir. Ayrıca, kümeleme sürecinin başlangıç değerlerine karşı oldukça duyarlıdırlar.

Bulanık kümeleme yöntemleri, diğeri kümeleme yöntemlerinin aksine veri uzayındaki her bir bireyin elde bulunan bütün kümelerle belirli ölçüde girmesine izin veren üyelik fonksiyonlarını kullanır. Bu üyelik değerleri, veri setinin ve kümelerin doğasını anlama açısından çok önemlidir. Hiyerarşik kümeleme gibi diğeri kümeleme yöntemleri sonuç olarak kesin kümeler verir ve bireylerin bu kümelerle üyelik dereceleri ya 0'dır ya da 1'dir. Oysa bulanık kümeleme de bireylerin üyelik dereceleri 0 ile 1 arasındadır ve bireylerin bütün kümelerle üyelik dereceleri toplandığında 1 elde edilir. Buradan üyelik dereceleri u_{ij} $i=1,2,\dots,n$ $j=1,2,\dots,c$ ise

$$u_{ij} > 0 \quad \forall i \text{ ve } \forall j \text{ için} \quad (4.1)$$

$$\text{ve} \quad \sum_{j=1}^c u_{ij} = 1 \quad (4.2)$$

şeklinde gösterilebilir.

Burada küme sayısı olan c kümeleme işlemi boyunca sabittir. Ancak c 'nin farklı değerleri denenip, en iyi sonucu veren küme sayılarından biri seçilebilir.

Diğer kümeleme yöntemlerine benzer olarak bulanık kümeleme de uzaklık ölçümüne dayanır. Bu yöntemin kullandığı uzaklık ölçümlerinin arasında Öklid ve Mahalanobis uzaklık ölçümleri yer alır. Bu uzaklık ölçümlerinden hangisinin seçileceği küme yapısına ve kullanılan algoritmaya bağlıdır. Bulanık kümelemenin kullanışlı bazı özellikleri vardır. Bunlar;

- Yorum açısından kullanışlı olan üyelik değerleri sağlar.
- Uzaklık kullanımı konusunda esnektir
- Üyelik değerlerinin bazıları bilindiğinde sayısal optimizasyon ile birleştirilebilir (Naes vd, 1999).

Bulanık kümelemenin geleneksel kümeleme yöntemlerine göre avantajı, veri hakkında daha detaylı bilgi vermesidir. Diğer taraftan dezavantajları da vardır. Çok sayıdaki birey ve küme durumunda çok fazla çıktı olacağından, özetlemek ve bilgiyi tasnif etmek zordur. Ayrıca bulanık kümeleme algoritmaları genellikle karmaşıktır ve daha çok belirsizlik söz konusu olduğunda kullanılır (Şahinli, 1999).

4.2 Geleneksel Bulanık Kümeleme Algoritmaları

Çoğu bulanık kümeleme algoritmaları amaç fonksiyonuna dayanmaktadır. Bu algoritmalar en iyi bölünmeyi amaç fonksiyonunu minimize ederek belirlerler. Amaç fonksiyonuna dayanan bulanık kümeleme algoritmalarında, her kümeyi bu kümenin prototipi temsil etmektedir. Bu küme prototipi, küme merkezi ve kümenin şekli, kümenin boyutu gibi bazı bilgileri içerir.

Verilen bir bireyin hangi kümeye gireceğini gösteren üyelik dereceleri, veri noktası ile küme merkezi arasındaki uzaklık hesaplanarak bulunur. Veri noktası hangi küme merkezine daha yakın ise, o kümeye ilişkin üyelik derecesi daha büyük olacaktır. Buradan anlaşılacağı gibi, bir veri setini c tane kümeye bölmek için veri noktaları ile küme merkezleri arasındaki uzaklığın minimize edilmesi ve üyelik derecelerine maksimize edilmesi gerekmektedir. Bu prensiplere dayanan birkaç kümeleme algoritması bulunmaktadır.

4.2.1 Bulanık c-ortalamlar algoritması (BCO)

Bulanık C-Ortalamlar algoritması, amaç fonksiyonuna dayanan bütün kümeleme tekniklerinin temelini oluşturmaktadır. İlk olarak Duda ve Hart (Duda vd, 1973) sert küme bölünmesini (Hard Cluster Partition) hesaplamıştır. Dunn ise (Dunn, 1974) bir bireyin birden fazla kümeye girebilmesini sağlamak için bu algoritmanın bulanık versiyonunu tanıtmıştır. Son olarak, Bezdek bulanıklık indeksini (m) dahil ederek algoritmanın son halini geliştirmiştir. BCO algoritması sonuçlandığında, p-boyutlu uzaydaki noktalar küresel bir şekil alır. Bu kümelerin yaklaşık olarak aynı boyutta olduğu varsayılır. Her bir kümeyi, küme merkezleri (centre) temsil eder ve bunlara prototip denir. Uzaklık ölçüsü olarak, veriler ile küme merkezi arasındaki Öklid uzaklığını kullanır.

$$d^2(v_i, x_k) = D_{BCO} = \|x_k - v_i\|^2 = \sum_{v=1}^p (x_k^{(v)} - v_i^{(v)})^2 \quad (4.3)$$

Burada $p \in N, D = R^p, \mathbf{x} = \{x_1, x_2, \dots, x_n\} \subseteq D, C = R^p, c \in N, R = P_c(C), m \in R_{>1}$ ve

$$d: D \times C \rightarrow R, (x, p) \rightarrow \|x - p\| \quad V = \{v_1, v_2, \dots, v_c\} \in R$$

Bu tekniğin uygulanabilmesi için, küme sayısının ve bireylerin kümeye üyelik derecelerinin önceden bilinmesi gerekmektedir. Bu tür parametrelerin önceden bilinmesi zor olduğundan, bu değerler deneme yanılma yoluyla ya da geliştirilen bazı teknikler ile bulunabilir.

Bu kümeleme yöntemi için kullanılan amaç fonksiyonu şu şekildedir:

$$J(u, v) = \sum_{j=1}^n \sum_{t=1}^c u_{t,j}^m \|x_j - v_t\|^2 \quad (4.4)$$

Bu fonksiyon en küçük kareler fonksiyonudur. n parametresi gözlem sayısını, c ise küme sayısını gösterir. u_{ij}^m t. kümedeki x_j 'nin üyeliği, J(u,v) değeri ise tüm ağırlıklandırılmış kare hatalarının toplamının bir ölçüsüdür (Şahinli, 1999).

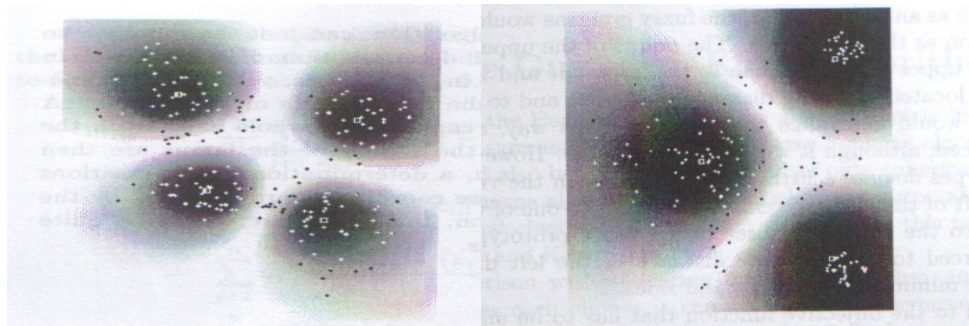
Eğer J(u,v) fonksiyonu c'nin her değeri için minimize edilecek olursa, diğer bir deyişle v_i 'lere göre 1.dereceden türevi alınıp 0'a eşitlenirse BCO Algoritmasının Prototipi şu şekilde olacaktır;

$$v_i = \frac{\sum_{j=1}^n u_{ij}^m x_j}{\sum_{j=1}^n u_{ij}^m} \quad (4.5)$$

olacaktır.

Burada bulanıklaştırıcı m BCO algoritması için önemli bir parametredir. Kaç tane kümenin üst üste geleceğini kontrol eder. m=1 (bulanıklık indeksi) seçildiği zaman BCO algoritması, Sert C-Ortalamlar (SCO) algoritmasının genelleştirilmiş bir biçimi olacaktır. Burada küme prototipleri yine aynı formül ile hesaplanır ve üyelikler ya 0 olacaktır ya 1 olacaktır. Veriler en küçük uzaklığa sahip olduğu kümeye girer. Ancak BCO algoritması 0'a bölüm hatasına yol açabileceğinden m değeri olarak 1 kullanılmasına izin vermez (Höppner, 1999). Eğer m değeri çok büyük seçilirse bireylerin kümelere etkileri çok küçük olacaktır ve bireylerin kümelere üyelik dereceleri yaklaşık olarak 1/c olacaktır.

BCO algoritmasında, küme prototipleri ile bireyler arasındaki uzaklık (4.3)'te verilen, sadece küresel şekle sahip kümeler için uygun olan Öklid uzaklık ölçüsü ile hesaplanır. Ancak oval, hat, dörtgen gibi farklı şekillere sahip küme çeşitleri bulunmaktadır ve hatta veri seti içindeki kümeler farklı boyuta ve yoğunluğa sahip olabilirler. BCO algoritması, küme boyutları ve yoğunluklarının farklı olduğu durumlarda da iyi çalışmamaktadır. Ayrıca küçük boyutlara sahip kümeleri de teşhis etmek zor olmaktadır. Bu tip problemleri çözmek için, Gustafson-Kessel, Bulanık C-Hatlar gibi bir takım algoritmalar geliştirilmiştir (Zhang, 2005).



Şekil 4.1 BCO Algoritması Sonucunda Elde Edilen Kümeler.

BCO algoritmasının uygulanması sonucunda oluşan kümeler Şekil 4.1'den görüldüğü gibi küresel yapıdadır.

BCO Algoritması için Gerekli Adımlar:

Adım1: Başlangıç Değerlerinin Girilmesi: Küme sayısı c , bulanıklık indeksi, m ve Üyelik dereceleri matrisi U veya V küme prototiplerini rasgele üret, işlem bitirme kriteri ϵ gibi,

Adım 2: V küme prototiplerinin rasgele üretildiği varsayılırsa bu değerler kullanılarak üyelik dereceleri matrisini hesapla

$$u_{ik} = \sum_{l=1}^c \left(\frac{\|v^i, x_k\|}{\|v^l, x_k\|} \right)^{-2/m-1} \quad (4.6)$$

Adım 3: (4.5) eşitliğine göre V küme prototiplerini güncelle

Adım 4: $\|V^{(t)} - V^{(t-1)}\| < \epsilon$ ise iterasyon durdurulur aksi takdirde Adım2'ye geri dönülür.

BCO algoritması uygulandıktan sonra hangi bireyin hangi kümeye gireceğine karar vermek için üyelik dereceleri kullanılır. Her bir bireyin hangi kümeye olan üyeliğinin en büyük olduğuna bakılır ve bu bireyler o kümeye dahil edilir. Ancak her bir birey diğer kümelere de belli bir üyelikle girerler.

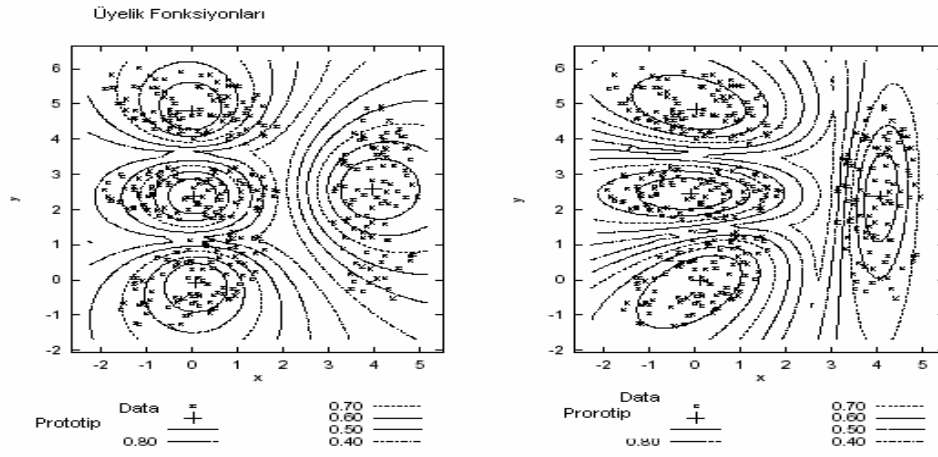
BCO algoritmasının sonucu başlangıçta rasgele üretilen değerlere oldukça bağlıdır. Bu yüzden bu rasgelelikten kaynaklanan problemleri ortadan kaldırmak için bir takım algoritmalar geliştirilmiştir.

4.2.2 Gustafson-Kessel algoritması (GKA)

Gustafson-Kessel Algoritması (GKA), küresel kümeler yerine, elipsoidal kümeleri teşhis etmek için geliştirilmiş bir bulanık kümeleme algoritmasıdır. Bulanık C-Ortalamlar algoritması bu tür kümeler için uygun değildir.

Bu algoritma, BCO algoritmasıyla karşılaştırıldığında, küme merkezlerine ek olarak her bir küme simetrik ve pozitif tanımlı bir A matrisine sahiptir. Bu matris her küme için $\|x\|_A = \sqrt{x^T A x}$ normuna sebep olur. Burada şu göz önüne alınmalıdır ki;

matrislerin keyfi olarak seçilmesi uzaklıkların küçük olmasına sebep olabilir. Amaç fonksiyonunun, girişleri yaklaşık olarak sıfır olan matris tarafından minimize edilmesini önlemek için, $\det(A)=1$ olacak şekilde sabit hacimli kümeler gerekir. Burada sadece küme biçimleri değişkendir, küme büyüklükleri değişken değildir. Gustafson-Kessel yöntemi, her bir A matrisi için sabit bir δ değeri tarafından ortaya çıkarılan farklı büyüklükteki kümelere izin verir. Ancak bu sabitin seçimi kümeler hakkında öncelikli bilgilerin olmasını gerektirir.



Şekil 4.2 (a) BCO Algoritması

(b) GK Algoritması

Şekil 4.2’de görüldüğü gibi, BCO algoritması, elipsoidal yapılara adapte olamamaktadır. Oysa; GKO algoritmasının dönüştürülmüş uzaklığı (Mahalanobis Uzaklığı), bu küme yapılarına da kolayca adapte olabilmektedir.

Gustafson Kessel Algoritmasının Prototipi:

$p \in N, D := R^p, X = \{x_1, x_2, \dots, x_n\} \subseteq D, C := R^p \times \{A \in R^{p \times p} | \det(A) = 1\}$ A simetrik ve pozitif tanımlı bir matristir, $c \in N, R := P_c(C), m \in R_{>1}$

$$d^2 : DXC \rightarrow R, (x, (v, A)) \mapsto (x - v)^T A (x - v) \text{ (Mahalanobis Uzaklığı)} \quad (4.7)$$

$$J(X, u, v) = \sum \sum u_{ik}^m (x_k - v_i)^T A_i (x_k - v_i) \quad (4.8)$$

$$A_i = \sqrt{\det(S_i)} S_i^{-1} \quad (4.9)$$

$$S_i = \sum u_{ij}^m (x_j - v_i)(x_j - v_i)^T \quad (4.10)$$

Gustafson-Kessel Algoritması S_i yerine, bulanık kovaryans matrisleri olarak adlandırılan aşağıdaki eşitliğe göre hesaplanan matrisleri kullanır.

$$F^i = \frac{\sum_{j=1}^n u_{ij}^m (x_j - v_i)(x_j - v_i)^T}{\sum_{j=1}^n u_{ij}^m} \quad (4.11)$$

Ancak $\frac{1}{\sum_{j=1}^n u_{ij}^m}$ sonucu etkilemez. Çünkü, matrisler determinantları 1 olacak şekilde ayarlanmıştır.

$J(X,u,v)$ amaç fonksiyonu minimize edildiğinde;

$$v_i = \frac{\sum_{j=1}^n u_{ij}^m x_j}{\sum_{j=1}^n u_{ij}^m} \quad (4.12)$$

(Höppner, 1999)

GK Kümeleme Algoritması için Gerekli Adımlar:

Adım1: Başlangıç Değerlerini Belirle:(c küme sayısı, i iterasyon sayısı, ϵ hata değeri, u üyelik değerleri vs gibi)

Adım 2: Bulanık küme merkezlerini her küme için hesapla (4.12)

Adım 3: Bulanık kovaryans matrisini her küme için hesapla (4.11)

Adım4: (4.7) formülünü kullanarak her bir birey için Mahalanobis uzaklığını hesapla

Adım5 : (4.6) formülünü kullanarak yeni üyelik değerleri matrisini hesapla

Adım6: Yeni üyelik değerleri ile eski üyelik değerlerini karşılaştır. $\|U^{(t)} - U^{(t-1)}\| < \epsilon$ ise iterasyonu durdur. Aksi takdirde Adım2'ye geri dön (Koçyiğit vd, 2005).

4.2.3 Gath-Geva algoritması (GGA)

Gath ve Geva Algoritması, kümelerin yoğunluğunu ve boyutlarını da dikkate alan Gustafson-Kessel Algoritmasının geliştirilmiş bir versiyonudur. Gerçekte bu yaklaşım, amaç fonksiyonu optimize etmeye dayanmaz. Gath-Geva algoritması, Maksimum Olabilirlik Tahmin Edicisinin bulanıklaştırılmasından elde edilen deneye dayalı bir metottur. Temel prensibi, veri noktalarının p-boyutlu normal dağıldığını varsaymaktır.

x_k veri noktaları, $k=\{1,2,\dots,n\}$, $N_i, i=\{1,2,\dots,c\}$ normal dağılımlar, klasik bir üyelik fonksiyonu düşünecek olursak $u_{ik} \in \{0,1\}$ Bu durumda istatistikten bilinen i . normal dağılımın ortalama değeri;

$$v_i = \frac{\sum_{k=1}^n u_{ik} x_k}{\sum_{k=1}^n u_{ik}} \quad (4.13)$$

ve uygun kovaryans matrisi

$$A_i = \frac{\sum_{k=1}^n u_{ik} (x_k - v_i)(x_k - v_i)^T}{\sum_{k=1}^n u_{ik}} \quad (4.14)$$

Bu eşitlikler Gustafson-Kessel algoritmasından elde edilen eşitliklerle benzerdir. Bulanık üyelik dereceleri için olasılık teorisinden sonuçların nasıl elde edileceği açıktır. Verilere göre elde edilen önsel olasılıklara P_i dersek;

x_j verisi için normalleştirilmemiş sonsal olasılık (olabilirlik) N_i normal dağılımı tarafından aşağıdaki gibi oluşturulur.

$$S^i = \frac{P_i}{(2\pi)^{p/2} \sqrt{\det(A^i)}} \exp\left(-\frac{1}{2}(x_j - v_i)^T A^{-1} (x_j - v_i)\right) \quad (4.15)$$

Gath-Geva algoritması için uzaklık fonksiyonu dolaylı olarak bu sonsal olasılık değerine orantılı olarak seçilir.

Gath-Geva Algoritmasının Prototipi:

$p \in N, D := R^p, X = \{x_1, x_2, \dots, x_n\} \subseteq D, C := R^p \times \{A \in R^{p \times p} \mid A \text{ pozitif tanımlı simetrik}\} \times R^c \in N R := P_c(C), m \in R_{>1}$

$$d^2(x, (v, A, P)) = \frac{1}{P} \sqrt{\det(A)} \exp\left(\frac{1}{2}(x-v)^T A^{-1}(x-v)\right) \quad (4.16)$$

Amaç fonksiyonunu yaklaşık olarak minimize edilirse;

$$v_i = \frac{\sum_{j=1}^n u_{ij}^m x_j}{\sum_{j=1}^n u_{ij}^m} \quad (4.17)$$

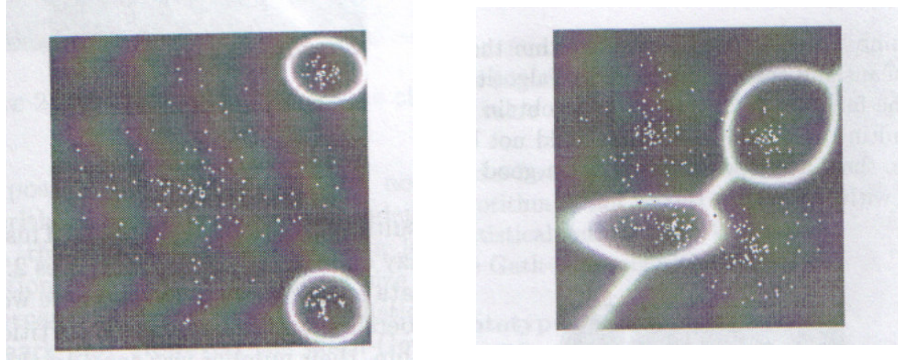
$$A_i = \frac{\sum_{j=1}^n u_{ij}^m (x_j - v_i)(x_j - v_i)^T}{\sum_{j=1}^n u_{ij}^m} \quad (4.18)$$

$$P_i = \frac{\sum_{j=1}^n u_{ij}^m}{\sum_{j=1}^n \sum_{i=1}^c u_{ij}^m} \quad (4.19)$$

Beklenen değer ve kovaryans matrisi için istatistiksel tahmin edicilerin geliştirilmesi, doğrudan küme merkezi v_i ve kovaryans matrisi A_i için hesaplama işlemlerine yol gösterir. P_i üyelikler için önsel olasılıkları tahmin eder. Normalleştirilmiş sonsal olasılıklar, i. normal dağılım tarafından oluşturulan bir verinin olasılığını belirtir. Uzaklık fonksiyonu da sonsal olasılıklara orantılı olarak seçilir. Eğer uzaklık küçük ise bunun anlamı üyelikler için yüksek olasılık, uzaklık büyük ise düşük olasılıktır.

Eğer Gath-Geva algoritması, BCO ve GK algoritmalarının kullanıldığı tekniğe göre uygulanırsa yani J amaç fonksiyonu minimize edilmeye çalışılırsa, prototipler için elde edilecek eşitlikler analitik olarak çözülemez. Bu bağlamda, Gath-Geva algoritması, olasılık teorisinin temelleri üzerine kurulmuş iyi bir deneysel çalışmadır. Gath-Geva algoritması, BCO ve GK algoritmalarının teşhis edebileceği

bütün örnekleri başarıyla kümeler. Ayrıca, düşük yoğunluklu ve geniş bir alana yayılmış kümeleri de kolayca teşhis eder. Uzaklık fonksiyonun içinde eksponansiyel fonksiyonun yer almasından dolayı, tüm uzaklıklar hemen hemen yakın ve uzak iki aralığa bölünür. Uzaklıklar, üyeliklerin 0 ile 1 den başka herhangi bir değer olması durumunda eksponansiyel fonksiyondan dolayı çok fazla artacaktır.



Şekil 4.3 GG Algoritmasının Uygulanması Sonucunda Elde Edilen Kümeler.

Gath-Geva Algoritması için Gerekli Adımlar:

Adım 1: Başlangıç değerlerinin girilmesi

Adım 2: Küme Merkezlerinin Hesaplanması (4.17)

Adım 3: Kovaryans Matrisinin Hesaplanması (4.18)

Adım 4: Önsel Olasılıkların Hesaplanması (4.19)

Adım 5: Uzaklıkların Hesaplanması (4.16)

Adım 6: Bölüm Matrisinin Güncellenmesi (4.6)

Adım 7: $\|U^{(t)} - U^{(t-1)}\| < \epsilon$ ise iterasyonu durdur. Aksi takdir de adım 2'ye geri dön.

4.2.4 Hesaplama zamanı

Klasik bulanık kümeleme algoritmalarını karşılaştırmak için, zaman indeksi τ verilebilir. Bu indeks hesaplama için gerekli olan t zamanının, n veri sayısı, c küme sayısı, i. iterasyon sayısının çarpımlarına bölümü ile elde edilir.

$$\tau = \frac{t}{c.n.i} \quad (4.20)$$

Bu zaman, aynı algoritmaların farklı analizleri için hesaplama zamanının sabit olmadığını gösterir. 10000 veri ve 10 kümenin olduğu durumdaki hesaplama zamanının her 10 saniyesi için, 1000 verinin ve 100 kümenin olduğu durumdakinden daha düşük bir zaman indeksi elde edilir. Bu karmaşık hesaplamalar her iterasyon adımında küme başına bir defa gereklidir. Bu yüzden 100 küme için harcanan toplam zaman elbette ki 10 küme için olandan daha büyük olacaktır. Küme sayısı, örneklerde çok fazla değişmez. Bu nedenle farklı analizler karşılaştırılabilir değerler ile sonuçlanır.

4.3 Prototipi Farklı Geometrik Şekle Sahip Kümeleme Algoritmaları

Bulanık kümeleme yöntemi ile veriler arasındaki doğrusal bağımlılıklarda teşhis edilebilir. Geleneksel regresyon tekniklerin aksine, birkaç küme çok karmaşık parça parça doğrusal ilişkileri temsil edebilir. Örneğin, bu doğrusal ilişkileri temsil etmek için fonksiyon tahmini için bulanık kural tabanlı yapılar oluşturulabilir. Ayrıca bulanık kümeleme yöntemleri ile daire çevresi, elips çevresi, hiperbol parabol şekilli küme prototipleri de teşhis edilebilir.

4.3.1 Bulanık c-regresyon (BCR) algoritması

Bu kümeleme yöntemi lineer prototipleri teşhis etmek için Bezdek tarafından geliştirilmiştir. Benzersizlik ölçüsü olarak klasik En Küçük Kareler yöntemine dayanan, lineerler arasındaki karesel uzaklığı kullanır ve aşağıdaki gibi tanımlanır.

$$d_{ik} = (y_k - f_i(x_k; \theta_i))^2 \quad (4.21)$$

Burada $x_k = [x_{k1}, x_{k2}, \dots, x_{kp}]$ k. veri noktasını, $\theta_i \in R^{p_i}$ ise lineer prototipin parametrelerini göstermektedir. Buna göre bu kümeleme yöntemine ilişkin amaç fonksiyonu aşağıdaki gibi tanımlanabilir:

$$J(U, \{\theta_i\}) = \sum_{i=1}^c \sum_{k=1}^N \mu_{ik}^m d_{ik}(\theta_i) \quad (4.22)$$

Burada c küme sayısını, m sonuç kümelerin bulanıklığını belirleyen bulanıklık indeksini, N ise toplam veri noktası sayısını göstermektedir. Bu amaç

fonksiyonu θ_i parametrelerine göre minimize edilerek lineer prototipler elde edilmektedir.

Buradan f_i lineer fonksiyon olduğundan $f_i(x_{ki}; \theta_i) = x_{ik}^T \theta_i$ şeklinde yazılabilir. x_{ki} 'ler x_k 'nin keyfi şekilli fonksiyonu olarak bilinir. Bu durumda parametreler bulanık bölünme matrisi U ile Ağırlıklandırılmış En Küçük Kareler probleminin bir çözümü olarak elde edilebilir.

N veri noktası ve üyelik dereceleri matrisi aşağıdaki gibi ayarlanabilir.

$$\mathbf{X}_i = \begin{bmatrix} \mathbf{x}_{i,1}^T \\ \mathbf{x}_{i,2}^T \\ \vdots \\ \mathbf{x}_{i,N}^T \end{bmatrix}, \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}, \mathbf{W}_i = \begin{bmatrix} \mu_{i,1} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mu_{i,2} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mu_{i,N} \end{bmatrix}.$$

Buradan en uygun θ_i parametreleri aşağıdaki gibi hesaplanabilir:

$$\theta_i = [X^T W_i X]^{-1} X^T W_i y \quad (4.23)$$

BCR Algoritması için Gerekli Adımlar:

Adım1: Başlangıç:

$2 < c < c_{\max}$ küme sayısını seç

Başlangıç Ayrışım Matrisini Rasgele Üret

ε işlem bitirme kriterini seç

$1 < m < \infty$ bulanıklık indeksini seç

Adım2: (4.23) eşitliğine göre parametreleri hesapla

Adım3 : Ayrışım Matrisini Güncelle

$$\mu_{ik} = \frac{1}{\sum_{j=1}^c (d_{ik}/d_{jk})^{2/(m-1)}} \quad 1 \leq i \leq c, \quad 1 \leq k \leq N \quad (4.24)$$

Adım4: $\|U^l - U^{l-1}\| \leq \varepsilon$ ise algoritmayı durdur. Aksi takdirde Adım2'ye geri dön.

4.3.2 Bulanık c-hatlar (BCH) algoritması

Bulanık C-Hatlar algoritması, hat, düzlem ve hiper düzlem gibi yapıları teşhis etmek için H.H.Bock (Bock, 1979) ve J.C.Bezdek (Bezdek vd, 1981) tarafından geliştirilmiştir. Her bir küme r boyutlu hatlar ile temsil edilir. Burada $r \in \{0, \dots, p-1\}$ veri uzayının boyutundan küçük olmak zorundadır ve p ise veri uzayının boyutudur. Her bir küme k_i , v_i noktası ve $e_{i,1}, e_{i,2}, \dots, e_{i,r}$ dikgen birim ile tanımlanan bir hiper düzlem ile temsil edilir.

$$v_i + \langle e_{i,1}, e_{i,2}, \dots, e_{i,r} \rangle = \{y \in R^p \mid y = v_i + \sum t_j e_{i,j}, t \in R^r\}$$

r=1 ise karışım düz bir hat, r=2 ise bir düzlem, r=p-1 ise karışım bir hiper düzlemdir. r=0 olması durumunda algoritma nokta- şekilli kümelerin teşhisine ve bulanık c-ortalama algoritmasına dönüşür. Burada r bütün kümeler için aynıdır. Bu algoritma, r'nin her küme için ayrı ayrı belirlenmesine izin vermez. Uygun uzaklık ölçüsü ise aşağıdaki gibidir.

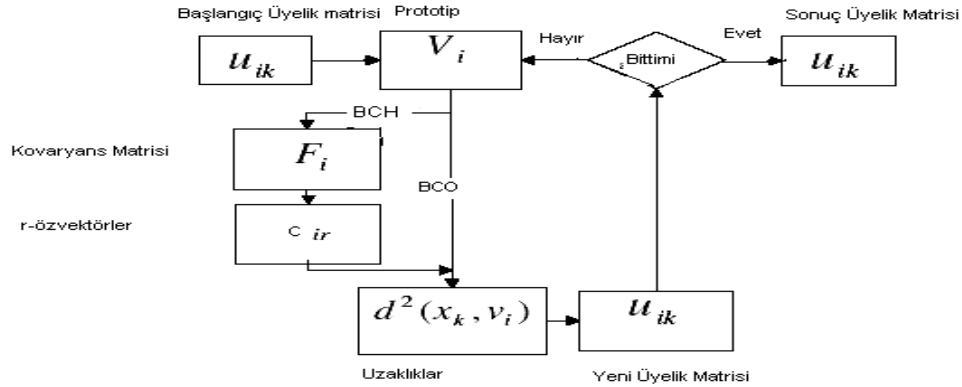
$$d^2(x, (v, (e_1, e_2, \dots, e_r))) = \|x - v\|^2 - \sum_{j=1}^r ((x - v)^T e_j) \quad (4.25)$$

Bu x vektörleri arasındaki Öklid uzaklık ve r- boyutlu hatlardır. r =0 olması durumunda bu uzaklık BCO algoritmasının kullandığı uzaklık fonksiyonuna dönüşür. Üyelik fonksiyonları verildiğinde BCH algoritmasının prototipleri (4.17) ve aşağıdaki gibidir.

$e_{i,l}$, C_i nin normalleştirilmiş özvektörü ise $l \in N_{<r}$ için

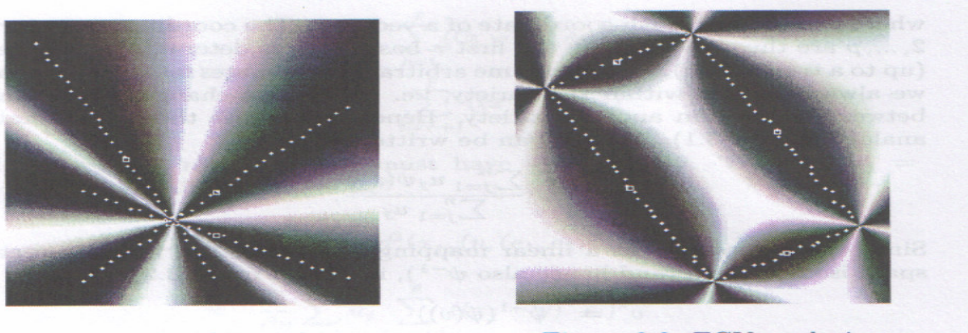
$$C_i = \frac{\sum_{j=1}^n u_{ij}^m (x_j - v_i)(x_j - v_i)^T}{\sum_{j=1}^n u_{ij}^m} \quad (4.26)$$

BCO ve BCH algoritmalarının arasındaki fark aşağıdaki şekilden görülmektedir.



Şekil 4.4 BCO ve BCH kümeleme algoritmaları.

Şekil 4.4'den de görüldüğü gibi, BCH algoritmasını elde etmek için BCO algoritmasına kovaryans matrislerinin ve r-özvektörlerinin hesaplanması dahil edilir.



Şekil 4.5 BCH Kümeleme Algoritması.

Şekil 4.5 (a) ve Şekil 4.5 (b) algoritmasının hatları nasıl teşhis ettiğini göstermektedir. Şekil 4.5 (a)'da kesişen 3 tane hat teşhis edilmiştir, (b)'de ise dikdörtgen şeklinde bir yapı oluşturmuş 4 tane hat görülmektedir.

Her iki şekilde de yüksek üyelik alanları, hat bölümlerinin (line segment) etrafında toplanmıştır. Her iki şekilde de hat bölümlerden çok uzakta veri

bulunmamaktadır. Bu durum kümeler üzerinde istenmeyen yanlı etkilerin olmasını engeller.

4.3.3 Uyarlamalı bulanık kümeleme (UBK) algoritması

Düz hatlar (straight line) yerine düz hat bölümlerini teşhis etmek için, uyarlamalı bulanık kümeleme algoritması geliştirilmiştir. Elliptotype merkezleri ve hat bölümleri üzerindeki noktalar arasındaki uzaklık dikkate alınarak, hat bölümleri farklı kümelere girerler. Bu algoritmada uzaklık ölçüsü olarak modifiye edilmiş BCH algoritması uzaklığı kullanılır.

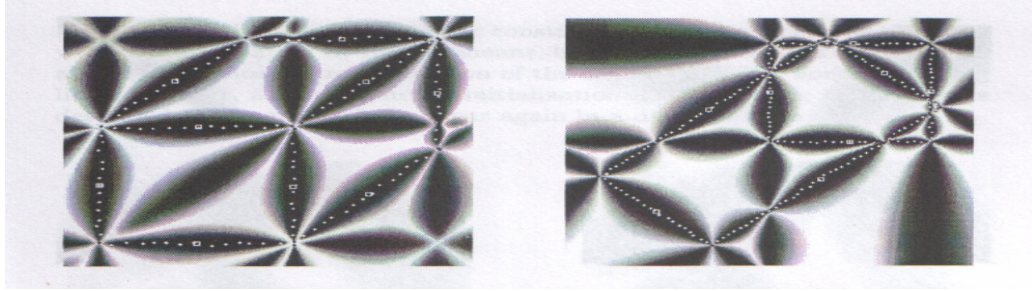
$$d^2(x, (v, e)) = \|x - v\|^2 - \alpha \sum_{j=1}^r (x - v)^T e_j \quad (4.27)$$

α 'nın seçimine göre küme şekilleri farklılık gösterir. $\alpha=0$ olması durumunda nokta şekilli küme $\alpha=1$ olması durumunda düz hatlar şekilli küme, $\alpha \in]0,1[$ olması durumunda ise eliptik şekilli küme olacaktır. α 'nın tüm kümeler için sabit olması durumunda, BCH algoritmasının bu modifikasyonu, Bulanık C-Elliptotype'lar (BCE) olarak bilinir (Bezdek, 1981). Dave, her bir küme için α 'nın seçimi ile ilgili bir öneride bulunmuştur (Dave, 1989). Her bir kümenin k_i ile gösterildiği ve C_i matrisinin özdeğerlerinin $\lambda_{i,1}, \lambda_{i,2}, \lambda_{i,3}, \dots, \lambda_{i,p}$ olduğu düşünülürse;

$$\alpha_i = 1 - \frac{\lambda_{i,p}}{\lambda_{i,1}} \quad (4.28)$$

Bu deneysel gelişmeler sadece 2-boyutlu durumda iyi sonuçlar vermektedir. Özdeğerler bulanık serpilme matrisinin (fuzzy scatter matrix), özvektörlerin yönüne doğru genişlemesi (expansion) hakkında bilgi sağlarlar. Doğrusal bir küme için, genişlemelerden biri, 0'a eşittir. Bu yüzden $\alpha = 1$ olur. Bu durumda, kümeler BCH algoritmasının uygulanması sonucunda elde edilen kümeler gibi davranır. Eğer ideal düz hat ele alınmıyorsa, en kısa ve en uzak genişleme arasındaki oran α değerini tanımlar. Özdeğerler ne kadar benzerse, kümeler o kadar küresel şekil almaktadır. Bu durumda α 0 yaklaşır ve kümeler BCO algoritması uygulamış gibi davranır. 3-boyutlu uzaydaki düz hatlar için bu yaklaşım başarılı bir şekilde çalışmaz. Veri vektörleri tamamıyla bir düzleme yerleşmişse, bu düzleme dikey genişleme 0 olacaktır. Böylece, özdeğerlerden biri 0 olur. Buradan α

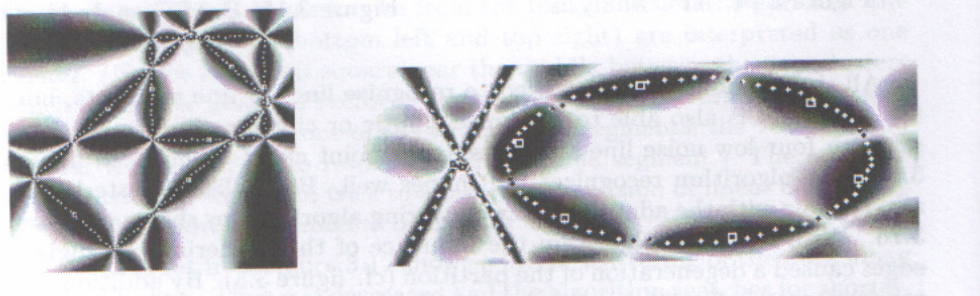
1'e yaklaşıyor ve algoritma, düzlem içindeki kümeler dairesel bir genişlemeye sahip olmasına rağmen ideal düz hatları araştırıyorlar.



Şekil 4.6 UBKA Uygulanması Sonucunda Elde Edilen Kümeler.

Şekil 4.6'daki küboid, uyarlamalı bulanık kümeleme algoritması tarafından teşhis edilir.

Bu algoritmaların dışında, Gustafson-Kessel ve Gath-Geva algoritmaları da hat yapıları içeren kümeleri teşhis etmede başarılı sonuçlar vermektedir. Gustafson-Kessel algoritması uygulandığı zaman ortaya çıkan kümeler aşağıdaki şekilden görülmektedir. Gath-Geva algoritması da Gustafson-Kessel algoritmasıyla yaklaşık sonuçlar vermektedir.



Şekil 4.7 Gustafson-Kessel Analizi.

GK algoritması, bu gibi verileri (hat bölümlerini) bölümlere ayırmada BCH algoritmasına göre daha iyi sonuçlar vermektedir ve BCH algoritması uygulanırken

karşılaşılan problemler ile karşılaşılmamaktadır. Öz değerler ve öz vektörleri belirlerken yüksek hesaplama zamanı gerektirmesine rağmen uyarlamalı bulanık kümeleme algoritmasına göre daha hızlıdır. Bu iki algoritma karşılaştırıldığında, GK algoritması uygulanması sonucunda oluşan kümeler daha dar görünür ve yüksek üyelik alanları daha küçük bir alana yayılmıştır. Nokta yoğunluğunu bu iki algoritma farklı farklı şekilde kümeler. GK algoritması ile düz hat kümeleri için aşırı uzatılan elipsler düz hat çizgilerine olan dikey uzaklığı oldukça değiştirir. Bu değişiklik görel olarak yakın olan noktaların bile düşük üyeliğe sahip olmasını sağlar. Bunlar daha sonra oval kümelere tahsis edilirler. Uyarlamalı bulanık kümeleme algoritmasında, bu etkiler meydana gelmez. Burada, bu noktalar doğru bir şekilde doğrusal kümelere tahsis edilirler (Höppner, 1999).

4.3.4 Kabuk (Shell) prototipler

Doğrusal olmayan eğri prototiplerini teşhis etmek için Kabuk Prototip (KP) Algoritmaları kullanılır. Daire çevresi, elips çevresi hatta hiperbol, parabol ve lineer yapıları teşhis etmek için kullanılır. c. küme için eğri prototipi şu şekilde tanımlanır:

$$x^T A_c x + x^T v_c + v_{co} = 0 \quad (4.29)$$

Burada A_c simetrik matris, (A_c, v_c, v_{co}) kümenin şeklini belirleyen parametrelerdir. Veri noktası x_k ile c. küme arasındaki matematiksel uzaklık

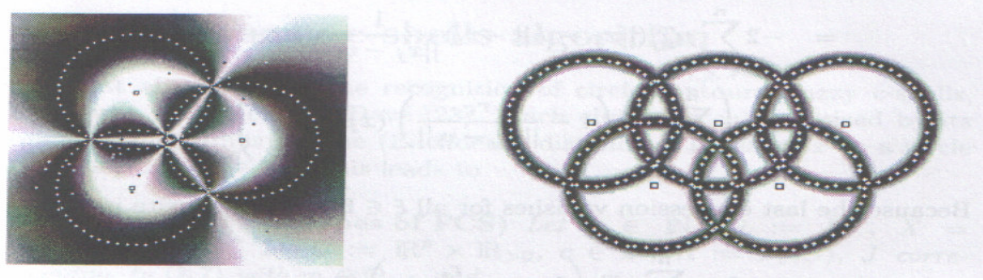
$$d_{ic}^2 = (x_i^T A_c x_i + x_i^T v_c + v_{co}) \quad (4.30)$$

Genel olarak KP algoritmalarının sonuç kümeleri başlangıç değerlerine oldukça bağlıdır. KP algoritmalarının başlangıç değerleri BCO algoritmalarından alınabilir. Buradan KP algoritmalarının bir derece küresel kümeleme yöntemleriyle ilgili olduğu anlaşılabilir.

Değişik KP algoritmaları bulunmaktadır. Örneğin daire çevresi şekilli kümeleri teşhis etmek için ilk olarak Dave tarafından **Bulanık C-Kabuk (BCK) Algoritması** geliştirilmiştir. Bu algoritmada her bir küme merkezi v ve yarıçap r ile tanımlanır. Herhangi bir x verisinin daire prototipine uzaklığı (Öklid) $\| \|x - v\| - r \|$ dir. Buradan amaç fonksiyonu aşağıdaki gibi yazılabilir.

$$J(u, v, r) = \sum_{j=1}^n u_{ij}^m (\|x_j - v_i\| - r_i)^2 \quad (4.31)$$

Ancak amaç fonksiyonu küme merkezi v ve yarıçapı r ye göre ayrı ayrı minimize edilecek olursa, küme merkezleri eşitlikleri kesin olarak çözülememektir. Bu durumda Newton iterasyonundan¹ yararlanılmaktadır.



Şekil 4.8 BCK Analizi.

Şekil 4.8'den BCK algoritması ile teşhis edilebilen kümeler gösterilmiştir.

BCK algoritmasının yüksek zaman karmaşıklığı (Newton algoritmasından dolayı) ve küme prototipleri eşitliklerinin kesin olarak çözülememesinden dolayı farklı uzaklık ölçülerini kullanan algoritmalar geliştirilmiştir. Bunlardan bir tanesi, **Bulanık C-Küresel Kabuklar (BCKK)** algoritmasıdır. Bu algoritma, için aşağıdaki uzaklık fonksiyonu kullanılmaktadır.

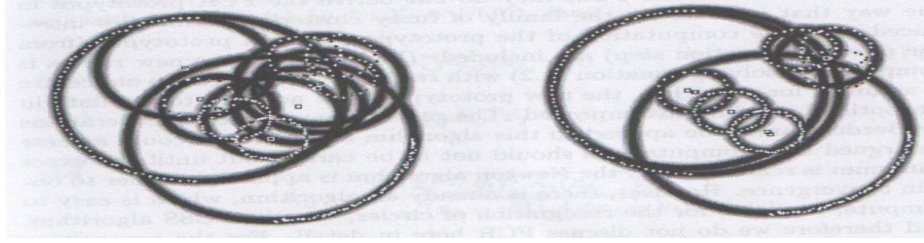
$$d^2 = (\|x - v\|^2 - r^2)^2 \quad (4.32)$$

Dolayısıyla buradan bu algoritma için amaç fonksiyonu

$$J(u, v, r) = \sum_{j=1}^n u_{ij}^m (\|x - v\|^2 - r^2)^2 \quad (4.33)$$

Bu algoritmanın başlangıç değerleri için genellikle BCO algoritmasının sonuçları kullanılmaktadır ve başlangıç değerleri küme sonuçları üzerinde oldukça etkilidir.

¹ Newton iterasyonu yinelemeli bir kök bulma algoritmasıdır.



Şekil 4.9 BCKK Analizi.

Bu algoritmaların dışında, daire çevresi şekilli ya da elips çevresi şekilli kümeleri teşhis etmek için Uyarlamalı Bulanık C-Kabuklar, Bulanık C-Elipsoitler Kabuklar, Bulanık C-Elipsler, Bulanık C-Quadratik Kabuklar gibi algoritmalarda mevcuttur.

4.4 Bulanık Kümelemede Doğrulama

Bulanık kümeleme uygun birkaç tane doğrulama göstergesi bulunmaktadır. Burada amaç, bir kümede olabildiğince çok, yüksek üyelik derecesine sahip elemanları, bir araya getiren kümeleme şemasını seçmektir. Bulanık kümele $[u_{ij}]$ şeklinde gösterilen bir U matrisi tarafından tanımlanır. Burada u_{ij} x_i vektörünün j . kümeye olan üyelik derecesini gösterir. Klasik kümelemeye benzer olarak burada da q doğruluk indeksinin m göre maksimum ve minimum olduğu noktalar atanır. q 'nun küme sayısına karşı bir trend göstermesi halinde, q 'nun önemli bir değişme gösterdiği nokta aranır.

Bulanık doğrulama göstergeleri için iki kategoriden bahsedilebilir. İlk kategoride, sadece bulanık bölünmenin üyelik değerleri kullanılır. İkinci kategori ise hem üyelik değerlerini hem de veri setini kullanır.

4.4.1 Sadece üyelik değerlerini kullanan doğruluk indeksleri

Bezdek tarafından önerilmiştir ve aşağıdaki gibi tanımlanır.

$$PC = \frac{1}{N} \sum_{i=1}^N \sum_{j=1}^{nc} u_j^2 \quad (4.34)$$

PC indeksi, $1/nc$ ile 1 arasında değerler alır. Burada nc küme sayısını göstermektedir. Bulanık bölünme sonucunda bütün üyelik değerleri eşit ise, $u_{ij} = 1/nc$ olacaktır ve PC'nin en küçük değeri elde edilecektir. PC'nin değeri, $1/nc$ 'ye yaklaştıkça kümeleme bulanıklaşacaktır. Ayrıca, $1/nc$ 'ye yakın bir değer, veri setinde kümeleme eğilimi olmadığını ya da kümeleme algoritmasının başarısız olduğunu gösterir.

Bölünme Entropi Katsayısı (Partition Entropy Coefficient): Bu indekste bu kategorideki bir başka doğrulama indeksidir ve aşağıdaki gibi tanımlanır:

$$BEK = -\frac{1}{N} \sum_{i=1}^N \sum_{j=1}^{nc} u_{ij} \log_a(u_{ij}) \quad (4.35)$$

Burada a logaritma tabanıdır. Bu indeks, 1'den büyük değerler için hesaplanır ve $[0, \log_a nc]$ aralığında değer alır. BE 0 yakın değer aldıkça kümeleme klasik kümeleme yaklaşır.

Bu göstergelerin bazı dezavantajları vardır:

- Monotonlukları küme sayısına bağlıdır. Bu yüzden, göstergenin küme sayısına göre çizilen grafiğinde, PC için önemli artma dizi (knee) BEK için önemli azalma dizi aranır.
- Bulanıklık indeksi m 'e karşı duyarlıdır. Özellikle m 1'ye yaklaştıkça nc 'nin tüm değerleri için aynı değeri alırlar.
- Verinin geometrik şekliyle direk bağlantıları yoktur.

4.4.2 Veri seti ve üyelik değerlerini kullanan doğruluk indeksleri

Xie-Beni indeksi, yoğunluk ve ayrılma doğrulama fonksiyonu olarak da bilinir. $X = \{x_j ; j=1, \dots, n\}$ veri seti v_i küme merkezleri, u_{ij} j . verinin i . kümeye giriş üyeliğini gösteren bir bulanık bölünme olsun.

x_j 'nin j . kümeye olan üyelik değerleriyle ağırlıklandırılmış uzaklık fonksiyonu:

$$d_{ij} = u_{ij} \|x_j - v_i\| \quad (4.36)$$

Bulanık kümelemenin ayrılması ise, küme merkezleri arasındaki minimum uzaklık olarak tanımlanır ve aşağıdaki gibi ifade edilir:

$$d_{\min} = \min \|v_i - v_j\| \quad (4.37)$$

XB indeksi ise aşağıdaki gibi tanımlanır.

$$XB = \pi / (N \cdot d_{\min}) \quad (4.38)$$

Burada $\pi = \sum_{i=1}^c \sum_{j=1}^N \mu_{ij}^m \|x_j - v_i\|^2$ verilerin küme merkezlerinden ağırlıklandırılmış toplam sapmalarını gösterir. N ise veri setindeki noktaların toplam sayısını göstermektedir. Yoğun ve iyi ayrılmış kümeler için XB değerlerinin küçük olması gerektiği açıktır. Ancak X değerleri, küme sayısı nc çok büyüdüğünde yani n'e yaklaştığında monoton olarak azalacaktır. Bu durumu ortadan kaldırmak için algoritmanın başında küme sayısı için c_{\max} maksimum bir değer verilir. Bundan başka XB'nin değerleri bulanıklık indeksi m'in değerine de bağlıdır. Bu kategoriye giren bir başka indeks ise Fukuyama-Sugeno indeksidir ve aşağıdaki gibi gösterilir.

$$FS_m = \sum_{i=1}^N \sum_{j=1}^{nc} u_{ij}^m (\|x_i - v_j\|_A^2 - \|v_j - v\|_A^2) \quad (4.39)$$

Burada v X'nin ortalama vektörüdür. A ise 1 x 1 boyutlu pozitif tanımlı, simetrik bir matristir. A birim matris olduğu zaman bu uzaklık fonksiyonu Öklid uzaklık fonksiyonuna dönüşecektir. Yoğun ve iyi ayrılmış kümelerin oluşabilmesi için FS_m değerinin küçük olması gerektiği açıktır. Parantez içindeki ilk terim kümelerin yoğunluğunu, ikinci terim ise küme temsilcileri arasındaki uzaklığı gösterir.

Bir başka doğruluk göstergesi ise Gath ve Geva (1989) tarafından önerilmiştir. Bu gösterge, yoğunluğa ve yüksek hacim (hypervolume) kavramlarına dayanır.

\sum_j j. kümenin kovaryans matrisini gösterir ve şu şekilde tanımlanır.

$$\sum_j = \frac{\sum_{i=1}^N u_{ij}^m (x_i - v_j)(x_i - v_j)^T}{\sum_{j=1}^N u_{ij}^m} \quad (4.40)$$

j. kümenin bulanık yüksek hacmi ise aşağıdaki gibi hesaplanır:

$$V_j = \left| \sum_j \right|^{1/2} \quad (4.41)$$

Burada $||$ determinant işaretidir. Bu ölçü küme yoğunluğunun bir ölçüsüdür.

Toplam bulanık yüksek hacmi ise aşağıdaki gibi tanımlanır:

$$FH = \frac{1}{nc} \sum_{j=1}^{nc} V_j \quad (4.42)$$

FH'in küçük değerleri yoğun kümelerin varlığını gösterir.

Ortalama bölünme yoğunluğu (average partition density) indeksi de bu kategoride yer alan indekslerdendir ve aşağıdaki gibi tanımlanır.

$$PA = \frac{1}{nc} \sum_{j=1}^{nc} \frac{S_j}{V_j} \quad (4.43)$$

Burada S_j j. kümeye giren veri noktalarının üyelik değerleri toplamını göstermektedir.

Farklı bir bölünme yoğunluk indeksi ise aşağıdaki gibi tanımlanır:

$$PD = S / FH \quad (4.44)$$

Burada S

$$S = \sum_{j=1}^{nc} S_j$$

*Küme içi homojenliği ve kümeler arasındaki heterojenliği sağlamak için geliştirilen bir diğer kümeleme doğrulama kriterleri ise V_{sv} Kim ve Park tarafından önerilmiştir. Bu en son kriter, eğer veri seti yoğun değilse ve iyi ayrılmamışsa tercih edilir.

Böylece, V_{sv} kriteri optimal küme sayısını bulma algoritmasına dayanır ve aşağıdaki gibi tanımlanır:

$$V_{sv} = V_u + V_o \quad (4.45)$$

Burada V_u , küme içi hatanın bir ölçüdür ve aşağıdaki gibi tanımlanır:

$$V_u = (1/n) \sum_{i=1}^c MD_i \quad 2 \leq c \leq c_{\max} \quad (4.46)$$

Burada MD_i küme içi uzaklığın ortalamasıdır ve aşağıdaki şekilde ifade edilir.

$$MD_i = \left(\sum_{x \in S_i} \|v_i - x\|^2 \right) / n_i \quad (4.47)$$

S_i i. kümeye giren verilerin kümesidir.

Kümeler arası hatayı belirlemek için ise aşağıdaki gibi ifade edilen V_o ölçüsü kullanılır (Kim vd, 1997).

$$V_o = c/d_{\min} \quad 2 \leq c \leq c_{\max} \quad (4.48)$$

d_{\min} kümeler arasındaki uzaklığının minimumudur.

$$d_{\min} = \min_{ij} \|v_i, v_j\| \quad (4.49)$$

5. BULANIK MODELLEME

5.1 Bulanık Modellemeye Giriş

Gerçek sistemlerin matematiksel modellerini oluşturmak birçok mühendislik ve fen bilimlerinin ana konusudur. Bu bilim dallarında, simülasyon, sistem hareketlerinin analizi, sistemi oluşturan mekanizmaların daha iyi anlaşılması, yeni süreçlerin tasarımı ya da denetleyicilerin tasarımı için modellerden faydalanılır.

Geleneksel olarak modelleme, sisteminin doğasının ve davranışının tam olarak anlaşılmasıyla uygun modelin ortaya çıkarılmasının birleşimidir. Bu yaklaşım genellikle “beyaz kutu” (fiziksel, matematiksel, ilk-prensip) modelleme olarak adlandırılır. Fakat karmaşık ve anlaşılması güç sistemler söz konusu olduğunda, eldeki problemin fiziksel alt yapısının iyi anlaşılması gerekliliği pratikte ciddi sınırlamalara sebep olur. Beyaz kutu modellemesi, temelde yatan olayların yeterince anlaşılabilmesi, çeşitli süreç parametrelerinin yanlış değerlere sahip olması ya da meydana gelen modelin karmaşıklığı gibi zorlukları ortaya çıkarabilir. Gerçek hayattaki birçok sistem için temeldeki mekanizmanın tam olarak anlaşılması neredeyse imkansızdır. Ayrıca, fiziksel modellemeyi yapabilmek için gerekli, yeterli derecede bilginin toplanması zor, zaman alıcı, pahalı ve hatta mümkün olmayabilir. Model yapısı belirlense bile parametreler için doğru değerlere ulaşmak büyük bir problem olarak kalabilir. Sistem üzerinde ölçülen değerlerden parametrelerin tahmin edilmesi sistem tanımlama işleminin görevidir. Tanımlama metodları, şu an için sadece doğrusal sistemlerin belirli seviyeleri için yapılabilmektedir. Fakat gerçekte birçok süreç doğrusal değildir ve kısmi olarak doğrusal modellere yaklaştırılarak tahmin edilebilirler.

Farklı bir yaklaşımda ise, üzerinde çalışılan süreç, genel fonksiyon yakınsayıcı olarak kullanılan genel “kara kutu ” yapısı kullanılarak tahmin edilir. Modelleme problemi, sistemin doğrusal olmayan yapısını ve dinamiğini korumak için, yakınsayıcının (tahminleyicinin) uygun yapısını gerçek olarak kabul eder.”kara kutu” modellemesinde, modelin yapısı zorda olsa gerçek sistemin yapısına yaklaştırılır. Tanımlama problemi, modelin parametrelerinin tahmin edilmesinden oluşur. Süreci temsil eden veriler elde edilebiliyorsa, “kara kutu” modellemesi süreç için herhangi bir ön bilgiye ihtiyaç duymadan, kolayca uygulanabilir. Bu

yaklaşımının en önemli dezavantajı, bu modellerin parametrelerinin ve yapısının fiziksel bir öneme sahip olmamasıdır. Bu modeller sistemin davranışını analiz etmekten çok sayısal simülasyonunu oluştururlar. Bundan başka, endüstride pratikte fazla kullanışlı değildirler.

“Beyaz kutu” ve “kara kutu” modellemenin avantajlarını birleştirmeye çalışan bazı modelleme teknikleri mevcuttur. Bu tür modeller genelde “melez” , “yarı mekaniktik” veya “gri kutu” modeller olarak adlandırılırlar. Çoğu standart modelleme tekniğinin uzman ve operatörlerin tecrübeleri gibi ekstra bilgileri göz ardı etmesi gibi dezavantajları vardır. İnsanoğlunun, yüksek belirsizliğe sahip karmaşık sistemlerin altından kalkabilme yeteneği, alternatif modelleme ve kontrol paradigmalarının araştırılmasını sağlamıştır. Bu araştırmalar sonucunda, dinamik sistemlerin kontrolü ve modellenmesi için geliştirilen, insan zekasından ve biyolojik sistemlerden esinlenen “akıllı” sistemler olarak adlandırılan teknikler ortaya çıkmıştır. Bu teknikler, doğal dil, kurallar, semantik ağlar ve nitel modeller gibi alternatif temsil şemalarını araştırırlar ve ilgili ekstra bilgiyi modellemeye dahil etmek için formal metotlara sahiptirler. Bulanık modelleme ve kontrol, insan bilgisinden ve çıkarsamalı ilgili süreçleri kullanan tekniklerin tipik örneğidir. Diğer taraftan, Yapay Sinir ağları biyolojik sinir sistemlerinin öğrenme ve adaptasyon yeteneklerini basit bir şekilde taklit ederek sonuca gitmeye çalışır. Tablo 5.1’de farklı modelleme paradigmaları özetlenmiştir.

Tablo5.1 Farklı Modelleme Paradigmaları.

Modelleme Yaklaşımı	Bilginin Kaynağı	Kullanılan metot	Örnek	Eksikliği
Beyaz kutu	Formal bilgi ve veri	Matematiksel	Diferansiyel Eşitlikler	Esnek (soft) bilgiyi kullanmaz
Kara kutu	Veri	Optimizasyon (öğrenme)	Regresyon sinir ağı	Tüm bilgiyi kullanmaz
Bulanık	Çeşitli bilgiler ve veri	Bilgi tabanlı ve Öğrenme	Kural-tabalı model	Boyutluluk problemi

5.2 Bulanık Modellemenin Ayırt Edici Özellikleri

Sistemlere ilişkin eksik ve belirsiz bilgiler: Geleneksel sistem teorisi, cebirsel ve diferansiyel ya da fark eşitlikleri gibi sistemin klasik matematiksel modellerine dayanırlar. Elektro-sistemler gibi bazı sistemler için matematiksel modeller elde edilebilir. Çünkü sistemi etkileyen fiziksel kanunlar bilinmektedir. Fakat uygulamadaki problemlerin birçoğu için fiziksel modeli kurmak için gereken bilgiyi toplamak zor, zaman alıcı, pahalı ve hatta imkansızdır. Birçok sistem için problem kısmi olarak anlaşılabilir, bu yüzden katı matematiksel modeller kurulamaz, kurulsa bile anlaşılmaları zordur. Bu tip sistemlerin örnekleri, kimya, gıda endüstrisi, biyoteknoloji, ekoloji, finans, sosyoloji gibi bilim dallarında görülebilir. Bu tür sistemler ile ilgili bilginin önemli bir bölümü, uzman kişilerden elde edilir ve bu bilgileri matematiksel olarak ifade etmek olanaksızdır. Çünkü elde edilen bilgi eksik ya da belirsizdir. Fakat sistemlerin işleyişini doğal dil, if-then kuralları şeklinde açıklamak mümkündür. Bulanık kural tabanlı sistemler, uzmanların konularıyla ilgili bilgi kullanılarak oluşturulan bilgiye dayalı modeller olarak kullanılabilirler (Pedrycz, 1990).

Kesin olmayan bilginin düzgün bir şekilde işlenmesi: Klasik matematiksel modelleri kullanarak yapılan kesin sayısal hesaplamalar parametrelerin ve girdi verisinin doğru bir şekilde bilindiği durumlarda işe yaramaktadır. Genelde sistem ile ilgili parametreler ve girdi verileri tam olarak bilinemediğinden, yalnızca bilinen veri ile değil, aynı zamanda belirsizliği de doğru bir şekilde ele alacak bir modelleme tekniğine ihtiyaç duyulur. Stokastik yaklaşım belirsizliği ele almanın geleneksel bir yoludur. Fakat bu yaklaşımında her türlü belirsizliği algılayamadığı ortaya çıkmıştır. Bu sebeple, yeni yaklaşımlar ortaya çıkmıştır. Bulanık mantık ve küme teorisi bunlardan biridir.

Gri kutu modellemesi ve tanımlanması: Girdi-çıkı verilerine göre dinamik sistemlerin tanımlanması, bilimde geniş bir uygulama alanına sahiptir. Gerçek hayatta karşılaştığımız birçok sistem doğası gereği doğrusal değildir ve klasik sistem tanımlama metotlarında kullanılan doğrusal modellerle temsil edilemezler. Son zamanlarda, doğrusal olmayan sistemleri ölçümün verilerinden tanımlamaya çalışan metotlara ilgi artmıştır. Yapay sinir ağları (YSA) ve bulanık modeller en popüler

yapılardır. Girdi-çıkıtı açısından bakılacak olursa, bulanık sistemler diğer fonksiyonları yaklaşık olarak tahmin eden esnek matematiksel fonksiyonlardır. Bu özelliğe genel fonksiyon yakınsayıcı adı verilir (Kosko, 1994). Yaygın olarak bilinen YSA gibi yaklaşım metotlarıyla karşılaştırıldığında, bulanık sistemler incelenen sistemin daha şeffaf bir görünümünü sunar. Bu özellikleri de kuralların uygun bir şekilde ele alınmasına bağlıdır. Kuralların mantıksal yapısı modelin anlaşılması ve analizinin yarı-niteliksel yani gerçek hayatı değerlendirdiği gibi ele alınmasını sağlar.

5.3 Bulanık Modellemenin Kullanım Alanları

Bulanık kümelerden ya da bulanık mantıktan faydalanan ve bunlara karşılık gelen matematiksel çerçeveyi kullanan statik ve dinamik sistemlere **bulanık sistemler** denir. Bulanık kümelerin bir sisteme dâhil edilmesinin birçok yolu vardır. Örneğin;

Sistemin tanımlanmasında: Bir sistem örneğin if-then kurallar yığınıyla (bulanık ifade) ya da bulanık ilişkilerle tanımlanabilir. Bir ısıtıcı ile odadaki sıcaklık değişimi arasındaki ilişkinin aşağıdaki şekilde tanımlanması bulanık kurallara örnektir.

If sıcaklık is yüksek then ısıtıcının derecesini düşür.

Sistem parametrelerinin belirlenmesinde: Bir sistemin parametreleri gerçek yerine bulanık sayılar kullanan cebirsel ve diferansiyel eşitlikler ile tanımlanabilir. Örnek olarak, $y = \tilde{3}x_1 + \tilde{5}x_2$ eşitliğinde $\tilde{3}$, yaklaşık olarak üç; $\tilde{5}$, yaklaşık olarak 5 anlamına gelen bulanık sayılardır. Bulanık sayılar parametre değerlerindeki belirsizliği ifade ederler.

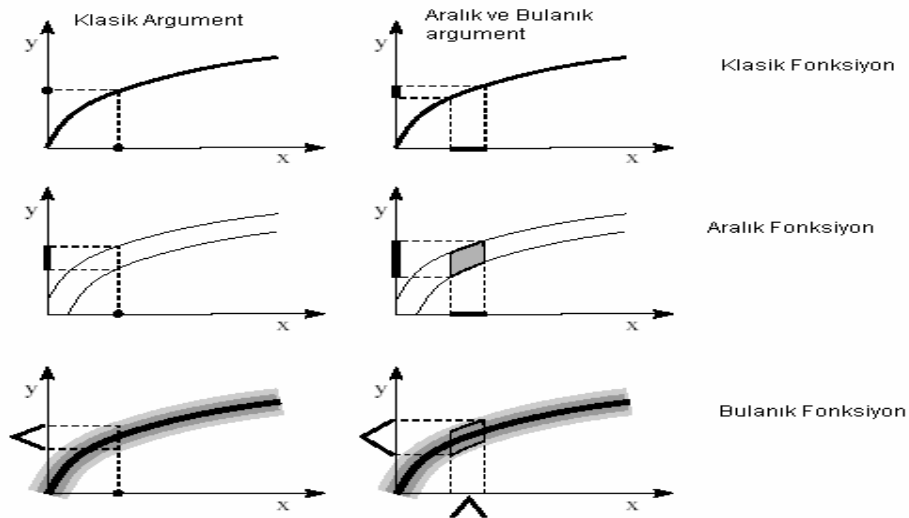
Sistemin girdi, çıktı ve durum değişkenlerinin bulanık küme olarak gösteriminde: Bulanık girdiler rahatlık ve güzellik gibi insan bakış açısıyla ilgili niteliksel bilgilerden elde edilebilir. Bulanık sistemler, klasik sistemlerin işleyemeyeceği bu tür bilgileri işleyebilirler.

Bulanık sistemler yukarıdaki özelliklere aynı anda sahip olabilirler. Tablo 5.2 bulanık ve klasik tanımlamalar ve değişkenler arasındaki ilişkiyi göstermektedir.

Tablo5.2 Sistemlerdeki klasik ve bulanık bilgi.

Sistem Tanımı	Girdi veri	Çıktı Veri	Matematiksel Temeli
Klasik (kesin)	Klasik	Klasik	Fonksiyonel analiz, lineer cebir vs
Klasik	Bulanık	Bulanık	Genişleme prensibi
Bulanık	Klasik Bulanık	Bulanık	Bulanık ilişkilerin hesaplanması, Bulanık çıkarm

Bulanık sistemlere, klasik sistemlerin geliştirilmiş bir biçimi dolayısıyla aralık-değerli sistemlerin geliştirilmiş bir biçimi gözü ile bakılabilir. Bu durum Şekil 5.1’de görülmektedir.



Şekil 5.1 Klasik, aralık ve bulanık argümanlar için, klasik aralık ve bulanık fonksiyonların değerlendirilmesi.

Burada, klasik, aralık ve bulanık veriler için fonksiyonların değerlendirilmesi şematik olarak gösterilmektedir. İlişki olarak, X ve Y'nin kartezyen çarpımının alt kümesi kullanılmıştır.

En yaygın bulanık sistemler if-then biçimindeki kurallar ile tanımlanırlar: Kural tabanlı bulanık sistemler. Bu bulanık sistemler, modelleme, veri analizi, tahmin ve kontrol gibi farklı amaçlar için kullanılabilirler (Babuska, URL3).

5.4 Bulanık Model Çeşitleri

5.4.1 Mamdani model

Mamdani tipi bulanık model insan davranışlarına çok uygun olduğundan dolayı çok kolay oluşturur. Bu nedenle yaygın bir kullanıma sahiptir ve diğer bulanık mantık modellerin temelini oluşturur. İlk defa bir buhar motorunun insan tecrübelerinden elde edilen sözel kontrol kuralları yardımıyla kontrolü amacıyla kullanılmıştır (Mamdani vd, 1975). Bu modelde hem girdi değişkenleri hem de çıktı değişkeni kapalı formdaki üyelik fonksiyonları ile ifade edilir (Akyılmaz, 2005).

Bu modelde, önce gelen kısım (Kuralın IF kısmı) ve sonra gelen kısım (Kuralın THEN kısmı) bulanık önermelerden oluşur.

$$R_i = \text{IF } x \text{ is } A_i \text{ THEN } y_i \text{ is } B_i \quad i=1, \dots, K \quad (5.1)$$

Burada A_i ve B_i bulanık kümelerle temsil edilen sözel terimlerdir (örneğin “küçük”, “büyük” gibi). K ise modelin kural sayısını göstermektedir. Sözel bulanık modeller niteliksel bilgiyi temsil etmek için oldukça kullanışlıdır.

Bu modelleme yönteminde modelleme işlemi uzman bilgisinden yararlanılarak yapılır. Yapısının tanımlanması (Girdi ve çıktı değişkenlerinin belirlenmesi, kural tabanındaki kuralların belirlenmesi, bulanık kümelerin bulunması) bilimsel temellerden daha çok uzman kişilerin görüşlerine dayanır.

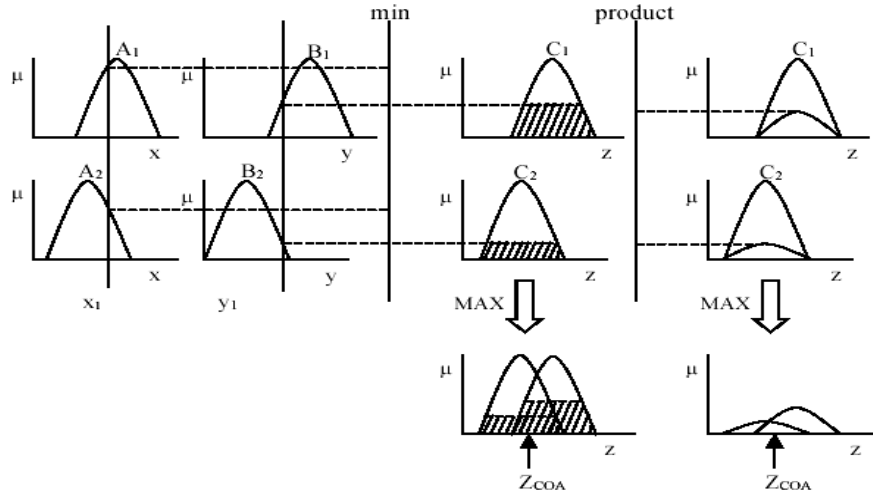
Mamdani tipi bir bulanık model aşağıdaki 5 adımda oluşturulur (URL1);

- a) Girdilerin bulanıklaştırılması: öncül kısımdaki bütün bulanık ifadeleri kullanarak girdi değişkenlerine ait 0 ile 1 arasında değişen üyelik derecelerinin belirlenmesi.
- b) Bulanık mantık işlemlerini kullanarak kural ağırlıklarının belirlenmesi.
- c) Bulanık küme mantıksal işlemcilerin (ve, veya) uygulanması.
- d) Sonuçların toplanması: her bir kuralın çıktısını temsil eden bulanık kümelerin birleştirilmesi.

e) Durulaştırma: tek bir sayıya dönüştürülmüş toplam bulanık küme sonuçlarının durulaştırılması.

Şekil 5.2’de iki girdi değişkeni ve tek çıktı değişkenine sahip Mamdani tipi bulanık modelin gösterimi görülmektedir. Mamdani tipi bulanık modellerde kuralları bileştirmek amacıyla max-min ve max-product işlemleri kullanılır. Kuralları birleştirmek için max-min kullanıldığında, her bir kuralın çıktısı girdi kümelerinin kesişimi olan bir bulanık küme olacaktır. Max-product kural birleştirme işleminde ise her bir kuralın çıktısı, cebirsel çarpım yoluyla küçültülmüş bir bulanık küme olacaktır. İki girdi değişkeni tek çıktı değişkeni olan Mamdani tipi bulanık model aşağıdaki gibi yazılır

IF x is A and y is B then z is C.



Şekil 5.2 Mamdani tipi bulanık model.

Şekil 5.2’den görüldüğü gibi iki girdi değişkeni tek çıktı değişkeni olan Mamdani tipi bulanık modelinde kuralları birleştirmek için max-min yöntemi kullanıldığında, kuralın çıktısı x ve y bulanık kümelerinin kesişim kümesinden oluşmaktadır. Ayrıca yine aynı şekilden, kuralları birleştirmek için max-product yöntemi kullanıldığı durumda, x ve y bulanık kümelerinin çarpılarak küçültüldüğü görülmektedir.

Mamdani bulanık modeller girdi olarak klasik(kesin) değerler kullanırlar ve bulanık kümeyi klasik değere dönüştürmek için durulaştırma işlemini kullanırlar.

Mamdani tipi bulanık modelin avantajlarını özetlemek gerekirse;

- Modelin oluşturulması basittir.
- Diğer bulanık mantık modellemenin temelini oluşturur.
- İnsan davranış ve duyularına uygundur (Yılmaz vd, 2005).

5.4.2 Takagi-Sugeno model

Takagi–Sugeno (Takagi, Sugeno, 1985) bulanık mantık yada Sugeno bulanık mantık ilk kez 1985 yılında kullanılmaya başlanmıştır. Mamdani bulanık mantık yönteminin bir uyarlamasıdır. Girdi değişkenlerinin bulanıklaştırılması ve bulanık mantık işlemleri Mamdani bulanık modelleme ile tamamen aynıdır. İki yöntem arasındaki fark çıktı üyelik fonksiyonlarındadır. Takagi-Sugeno tipi bulanık modellemede çıktı üyelik fonksiyonları lineerdir (URL1).

Bir başka deyişle bu modelleme yaklaşımında, model bulanık ifadelerden ve kurallar tarafından oluşturulur. Kuralın IF kısmı bulanık değişkenleri içeren bulanık ifadeleri, her bir bulanık ifadenin THEN kısmı ise modelin girdi değişkenlerinin parametrelerden oluşan eşitliğini gösterir. i. bulanık ifade için R^i

$$R^i = \text{IF } x_1 \text{ is } A_1^i \text{ and } x_2 \text{ is } A_2^i, \dots \text{ and } x_k \text{ is } A_k^i \text{ then } y^i = a_0^i + a_1^i x_1 + \dots + a_k^i x_k \quad (5.2)$$

Burada R^i ($i=1,2,\dots,c$) bulanık kuralı, $x_j(j=1,2,\dots,k)$ girdi değişkenlerini, y^i bulanık kuralın çıktısını, a_j^i 'ler ise modelleme sürecinden elde edilen model parametrelerini göstermektedir. Modelin yapısı girdi değişkenlerin sayısı, bulanık değişkenlerin sayısı veya her bir girdi değişkenin üyelik değerleri ile belirlenir. Modelin yapısı, modellenecek sistem hakkındaki ön bilgiye dayanarak önceden belirlenebilir.

Basit olarak bu sistem çok-girdili-tek-çıkıtlı sistem olarak varsayılır. Çok çıktı olması durumunda, her kural için $y_1^i, y_2^i, y_3^i, \dots, y_n^i$ şeklinde çok sayıda çıktı değişkeni olur. Ayrıca kuraldaki $A_1^i, A_2^i, \dots, A_k^i$ değişkenleri, bulanık alt uzayı temsil

eden üçgen, yamuk, çan eğrisi veya diğer üyelik fonksiyonları ile birlikte kullanılan bulanık değişkenlerdir

Bulanık modelin çıktısı \hat{y} , aşağıdaki gibi gösterilir.

$$\hat{y} = \frac{\sum_{i=1}^c w^i y^i}{\sum_{i=1}^c w^i} \quad w^i = \text{MIN}_{j=1}^k A_j^i(x_j) \quad (5.3)$$

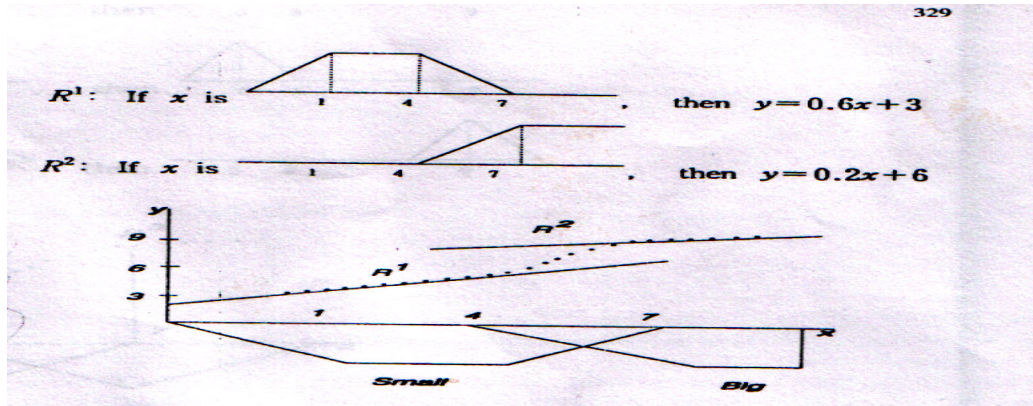
Burada c kural sayısını, k ise girdi değişkenlerinin boyutunu göstermektedir.

T-S dinamik ve statik lineer olmayan sistemleri modellemek için kullanılır ve girdi uzayının bulanık bölünmesine dayanır. T-S bulanık model Şekil 5.3te görüldüğü gibi lineer olmayan sistemleri birkaç farklı lineer sistemin kombinasyonu olarak varsayar. Bulanık modelleme yapılırken ilk önce kuralın sol tarafındaki öncül parametreler (premise parameter) elde edilir daha sonra, verilen öncül parametrelere göre kuralın sağ tarafındaki soncul parametreler (consequent parameter) ayarlanır. Bu işlem kompleks algoritma tarafından iteratif olarak devam eder.

İki kurala sahip T-S Modeli:

$$R^1: \text{IF } x \text{ is } A_1 \text{ THEN } y=0.6x + 3$$

$$R^2: \text{IF } x \text{ is } A_2 \text{ THEN } y=0.2x + 6$$



Şekil 5.3 T-S Bulanık Model.

Şekil 5.3'ten görüldüğü gibi, her bir kuralın çıktısı lineer bir fonksiyonu göstermektedir. Bu kuralları birleştirmek amacıyla, her bir lineer fonksiyon girdi uzayından elde edilen üyelik değerleriyle ağırlıklandırılır.

$$Y(\text{Çıktı}) = (A_1 \cdot (0.6x+3) + A_2 \cdot (0.2x+6)) / (A_1 + A_2)$$

T-S tipi bulanık modelin avantajları aşağıda sıralanmıştır (URL1; URL2);

- Lineer olmayan sistemlerin kontrol edilmesi için lineer teknikler kullanılabilir.
- Optimizasyon ve uyarlanabilir (adaptive) tekniklerle birlikte iyi çalışır ve çıktı parametrelerini optimize ederek sonuçları iyileştirir.
- Çıktı uzayında sürekliliği garantiler.
- Matematiksel analiz için uygundur.
- Hesaplama için çok uygundur.

T-S tipi bulanık modelin dezavantajları ise (URL1);

- Yüksek derecedeki Sugeno bulanık modelleme kullanıldığında oldukça kompleks bir yapıya sahip olur.
- Girdi ve alt küme sayılarının artması verilerin eğitilmesini zorlaştırır, sonuçların elde edilmesi için belirlenmesi gereken soncul parametrelerin sayısı artar.
- İnsan sezgilerine çok uygun değildir.

5.4.3 Sugeno-Yasukawa model

Bu bulanık model 1993 yılında Sugeno ve Yasukawa tarafından önerilmiştir. T-S bulanık modele göre bazı avantajlara sahiptir. Bunlar;

- Bu modelde soncul kısım sözel değişkenler ile ifade edilir ve T-S bulanık modele nazaran daha çok sezgiye dayanır.
- Bu modelin uygulanması T-S modelden daha kolaydır.

Bu model aşağıdaki gibi temsil edilir.

$$R^i = \text{If } x_1 \text{ is } A_1^i \text{ and } x_2 \text{ is } A_2^i, \dots, \text{ and } x_m \text{ is } A_m^i \text{ then } y^i = B^i \quad (5.4)$$

Çıkarım ise aşağıdaki gibi yapılır:

$$\tilde{y} = \frac{\sum_{i=1}^c w^i b^i}{\sum_{i=1}^c w^i} \quad (5.5)$$

Burada $b^i = \int yB^i(y)dy / \int B^i(y)dy$ soncul üyelik fonksiyonunun alan merkezidir.

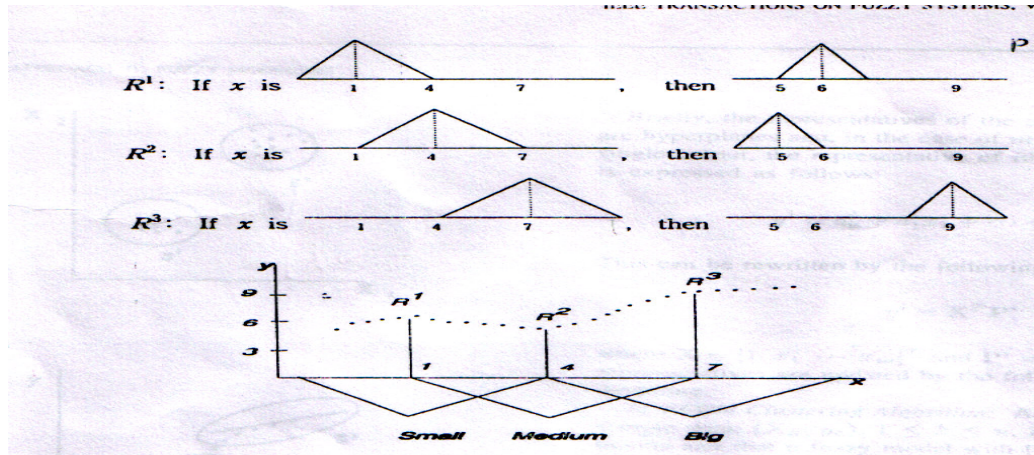
$w^i = \text{MIN}_{j=1}^m A_j^i(x_j)$ ve $A_1^i, A_2^i, \dots, A_m^i, B^i$ ($i=1,2,\dots,c$) çan eğrisi üçgen veya yamuk şekilli bulanık değişkenlerdir. Sugeno-Yasukawa bulanık modelde parametreler kabaca BCO algoritmasından ve kompleks algoritmanın optimize edilmesinden elde edilir. Parametrelerin basit bir şekilde elde edilmesi Sugeno-Yasukawa algoritmasını T-S'dan daha basit yapar (Kim vd, 1997).

Şekil 5.4'e bakıldığında Sugeno-Yasukawa modeli için kurallar aşağıdaki gibi yazılabilir.

R^1 : IF x is A_1 THEN y = 6

R^2 : IF x is A_2 THEN y = 5

R^3 : IF x is A_3 THEN y = 9



Şekil 5.4 Sugeno-Yasukawa Bulanık Model.

Şekil 5.4'den görüldüğü gibi, her bir kuralın çıktısı tek bir sayısal değere karşılık gelmektedir. Bu kurallar aşağıdaki gibi birleştirilir.

$$Y(\text{Çıktı}) = (A_1*6 + A_2*5 + A_3*9)/(A_1 + A_2 + A_3)$$

5.4.4 Anahtarlamalı bulanık regresyon model

$S = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$ şeklinde bağımsız değişken x_k ve bağımlı değişken y_k sahip bir veri seti verildiğinde, bunlar arasındaki ilişkiyi bulmanın en kolay yolu tek bir fonksiyonel ilişki kullanmaktır. Çoğu durumda istatistiksel teknikler bu ilişkinin seçimini ölçülen hatalara göre yapmaktadır. En uygun fonksiyonun seçimi aşağıdaki gibi varsayılan ilişki ile sınırlandırılır (Hathaway vd, 1993).

$$y = f(x; \beta) + \varepsilon \quad (5.6)$$

Burada β belirlenmesi gereken parametre vektörünü ε ise 0 ortalamaya ve Σ kovaryans matrisine sahip rasgele bir vektörü göstermektedir. Burada en uygun β tanımlanması, ε hakkındaki dağılıma ilişkin varsayımlara bağlıdır. Bu model çok değişkenli istatistikte sıkça kullanılmaktadır.

Son zamanlarda bağımlı ve bağımsız değişken arasındaki ilişkiyi tanımlamada bulanık teoriye dayanan yöntemlerde kullanılmaktadır. Bu yaklaşımlardan biri olan Anahtarlamalı Bulanık Regresyon Model'de, bu değişkenler arasındaki ilişki tek fonksiyonel ilişki ile değil çoğul fonksiyonel ilişki ile tanımlanmaktadır. (5.6) eşitliği çoğul fonksiyonel ilişki olarak kullanıldığı durumda aşağıdaki gibi olacaktır:

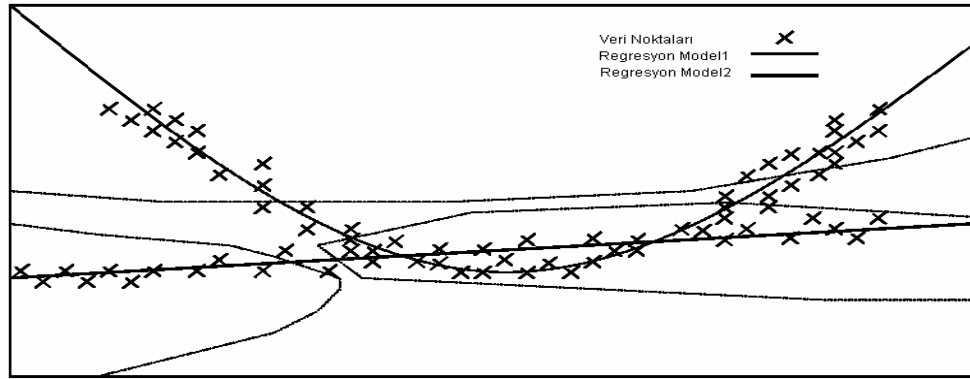
$$y = f_i(x; \beta_i) + \varepsilon_i \quad 1 \leq i \leq c \quad (5.7)$$

Burada β_i her bir parametre vektörünü, ε_i ise $\mu_i = 0$ ortalama ve Σ_i kovaryans matrisine sahip rasgele vektörü göstermektedir. $\{\beta_1, \beta_2, \dots, \beta_c\}$ parametreleri için en iyi tahmin tek fonksiyonel ilişki durumunda bulunduğu gibi elde edilmektedir. Ancak buradaki problem herhangi bir (x_k, y_k) veri noktası için hangi modelin seçileceğidir. Bunun için bulanık kümeleme teknikleri kullanılabilir. Bu amaç doğrultusunda, en uygun regresyon modellerinin tanımlanması için S veri setinin bölünmesini ve $\beta_1, \beta_2, \dots, \beta_c$ parametrelerin belirlenmesini eş zamanlı olarak

yapan bir kümeleme tekniği araştırılmalıdır. Bu durumda, Bölüm 4'te de değinilen BCR algoritmasının aşağıdaki gösterilen amaç fonksiyonunun bu amaç için uygun olduğu görülmektedir.

$$J(U, \{\beta_i\}) = \sum_{i=1}^c \sum_{k=1}^N \mu_{ik}^m d_{ik}(\beta_i) \quad (5.8)$$

BCR kümeleme algoritması kullanılarak S veri setinin bölünmesi ve parametrelerin belirlenmesi işlemi yapıldıktan sonra, (x_k, y_k) veri noktası için hangi regresyon modelinin seçileceğine karar verilmesi gerekmektedir. Herhangi bir veri noktası için regresyon modeli seçerken ayrışım matrisi kullanılmaktadır. Bu durumda ayrışım matrisinden yararlanarak veri noktasının hangi kümeye üyeliği daha fazla ise o veri noktası için o kümeye karşılık gelen regresyon modeli seçilmektedir. Bu işleme Anahtarlama işlemi denilmektedir (Hathaway vd, 1993).



Şekil 5.5 Anahtarlama.

Şekil 5.5'ten görüldüğü gibi veri noktaları hangi küme prototipine daha yakın ise, başka bir deyişle hangi kümeye olan üyeliği daha fazla ise o kümenin regresyon modelini kullanmaktadır.

$f_i(x_k; b_i)$ olarak doğrusal denklem kullanıldığında model şöyle tanımlanacaktır:

$$y_i = b_{0i} + b_{1i}x_1 + b_{2i}x_2 + \dots + b_{pi}x_n + \varepsilon_i \quad (5.9)$$

Bu durumda küme prototipi de Şekil 5.5 regresyon model 2’de görüldüğü gibi lineer olacaktır.

$f_i(x_k; b_i)$ doğrusal olabileceği gibi quadratik formda da olabilir. Bu durumda da kullanılacak amaç fonksiyonu (5.8)’de olduğu gibidir ve algoritma da aynı şekilde uygulanır. Bu durumda küme prototipi Şekil 5.5 regresyon model 1’dekine benzer biçimde eğrisel olacaktır.

Anahtarlamalı Bulanık C-Regresyon Algoritması adımları ile sırasıyla aşağıdaki verilmiştir.

- S veri setinin bölünmesi ve regresyon modellerinin bulunması.

Adım 1: Başlangıç değerlerinin girilmesi:

Küme sayısı $2 < c < c_{\max}$

İşlem Bitirme kriteri ε seç

Bulanık indeksini seç $1 < m < \infty$

Başlangıç Ayrışım Matrisini Rasgele Seç

Adım 2: (5.8) eşitliğini minimize edecek b_i parametrelerinin hesaplanması

Adım 3: Ayrışım Matrisinin Güncellenmesi:

$$\mu_{ik} = \frac{1}{\sum_{j=1}^c (d_{ik}/d_{jk})^{2/(m-1)}} \quad 1 \leq i \leq c, \quad 1 \leq k \leq N \quad (5.10)$$

Adım 4: $\|U^l - U^{l-1}\| \leq \varepsilon$ ise algoritmayı durdur. Aksi takdirde Adım2 ‘ye geri dön.

- Tahmin

$$y_i : x_k / u_{ik} > u_{jk}, i \neq j, i = 1, \dots, c, k = 1, \dots, n \quad (5.11)$$

6. BULANIK KÜMELEMeye DAYANAN BULANIK MODELLERİN GELİŞTİRİLMESİ

6.1 Geliştirilmiş Takagi-Sugeno Model

Takagi-Sugeno bulanık modelleme yönteminin uygulanabilmesi için önceden, modelin yapısının, kural sayısının ve başlangıç ayrışım matrisinin belirlenmesi gerekmektedir. Daha önceki çalışmalarda (Takagi, Sugeno, 1985) kural sayısının belirlenmesi ve modelin yapısının belirlenmesi işlemi, soncul kısmın optimal olarak ayarlanabilmesi için lineer olmayan optimizasyon algoritması gerekmektedir. Bu model yapısının belirlenmesi prosedürü bulanık kuralın hem öncül kısımdaki hem de soncul kısımdaki parametreler için optimal sonuç elde edilene kadar devam etmesi gerekmektedir. Bu işlemler hesaplama açısından çok zaman alıcı ve uygulanması karmaşık olan işlemlerdir. Bu gibi dezavantajlar dikkate alınarak (Eminov vd, 2004) çalışmasında olduğu gibi yapının tanımlanması ve parametrelerin belirlenmesi iki aşamada yapılabilir. Yapının tanımlanması adımında hem kuralın öncül kısmındaki bulanık kümeler, hem de kuralların uygun sayısı elde edilir. Parametrelerin belirlenmesi adımında ise yapının tanımlanması adımında elde edilen bulanık kuralların soncul kısmındaki parametreler elde edilir (Eminov vd, 2004).

6.1.1 Model yapısının tanımlanması

Adım 1: Bulanık Kuralların Belirlenmesi.

Takagi-Sugeno (T-S) bulanık modelin yapısının aşağıdaki gibi olduğu Bölüm 5'ten bilinmektedir.

$$R^i = \text{IF } x_1 \text{ is } A_1^i \text{ and } x_2 \text{ is } A_2^i, \dots \text{ and } x_k \text{ is } A_k^i \text{ then } y^i = a_0^i + a_1^i x_1 + \dots + a_k^i x_k \quad (6.1)$$

Bu model için, yapı tanımlama adımında sadece kural sayısı ve kuralların öncül kısmındaki parametrelerin belirlenmesi ile ilgilenir. T-S bulanık modeli aşağıdaki gibi geliştirilirse:

$$R^i : \text{ if } x \text{ is } A^i \text{ then } y^i = b_0 + \sum_{j=1}^p b_j^i x_j \quad (6.2)$$

Burada $x = (x_1, x_2, \dots, x_p)$ çok boyutlu girdi değişkeni, $A^i = (A_1^i, A_2^i, \dots, A_p^i)$ çok boyutlu $A^i(x_1, x_2, \dots, x_p)$ üyelğine sahip bulanık alt kümeyi göstermektedir. Bu parametreler kullanılarak, öncül kısımdaki (Kuralın IF kısmı) bulanık alt kümelerin şekli üyelik fonksiyonları tarafından tanımlanabilir. Her bir bulanık kümenin bir bulanık alt küme olduğu düşünülürse, girdi uzayı için üyelik fonksiyonu aşağıdaki gibi tanımlanabilir (Kim vd, 1997).

$$A^i(x) = \exp\left(\frac{\sum_{j=1}^p (x_j - v_j^i)^2}{2(\delta^i)^2}\right) \quad (6.3)$$

Burada $V^i = (v_1^i, v_2^i, \dots, v_p^i)$ $i=1, \dots, c$ çok boyutlu küme merkezi, δ^i ise i . kümenin standart sapmasını göstermektedir.

Adım 2: Girdi Uzayının BCO Algoritması Kullanılarak Bölümlemesi.

V^i ve δ^i gibi parametrelerini belirlemek için BCO algoritması kullanılabilir. Bu algoritmaya göre veri setinin c sayıda kümeye bölünmesi, U üyelik dereceleri matrisi kullanılarak elde edilir. Girdi uzayını c sayıda bölgeye ayırmak için aşağıdaki amaç fonksiyonunun minimize edilmesi gerekmektedir.

$$J(U, V) = \sum_{i=1}^c \sum_{k=1}^n (u_{ik})^m \left\| V^i, X_k \right\|^2 \quad (6.4)$$

Bu amaç fonksiyonunun optimizasyonu için, başlangıçta sabit bir küme sayısı kullanılır ve başlangıç ayrışım matrisi $U_{(0)}$ veya küme merkezleri V_o^i rasgele üretilir. Her iterasyon adımında bu değerler aşağıdaki eşitliklere göre güncellenir.

$$u_{ik} = \frac{\left\| V^i, X_k \right\|}{\sum_{l=1}^c \left\| V^l, X_k \right\|} \quad (6.5)$$

$$V_i = \frac{\sum_{k=1}^n (u_{ik})^m X_k}{\sum_{k=1}^n (u_{ik})^m} \quad (6.6)$$

Bu adımlar işlem bitirme kriteri ϵ sağlanana kadar tekrar edilir. Yapının tanımlanması adımında üyelik değerlerinin belirlenmesinden önce uygun kural sayısının belirlenmesi gerekmektedir.

Adım 3: Uygun Kural Sayısının Belirlenmesi.

Bulanık kuralların sayısı veri setine ilişkin küme sayısına eşittir ve dolayısıyla kuralların oluşturulabilmesi için çok önemlidir. Uygun küme sayısı doğruluk indeksleri kullanılarak elde edilebilir. Bu amaçla, doğruluk indekslerinden 2001 yılında Kim ve Park tarafından geliştirilen V_{sv} doğruluk indeksi kullanılabilir.

Bunun için geliştirilen algoritma aşağıdaki gibidir;

- Başlangıç: minimum ve maksimum küme sayılarının gir
- BCO algoritması kullanılarak verileri kümelere tahsis et
- Bölüm 4'te değinilen V_u (4.46) ve V_o (4.48) değerlerini hesapla
- V_{sv} (4.45) değerini hesapla
- Küme sayısını 1 artır $c \leftarrow c+1$
- Maksimum küme sayısına ulaşıldığında işlemin bitir
- Minimum V_{sv} değerine sahip olan küme sayısını uygun küme sayısı olarak seç

Adım 4: Üyelik Değerlerinin Belirlenmesi

Yapı tanımlanması işlemin son olarak üyelik fonksiyonları belirlenir. Bunun için (6.3)'te tanımlanan üyelik fonksiyonun bulunması gerekmektedir. Uygun küme sayısı, girdi uzayının üyelik değerleri ve küme merkezleri bulunduktan sonra bulunması gereken bir diğer parametre ise δ^i 'dir. δ^i belirlemek için kullanılan eşitlik aşağıdaki gibidir:

$$\delta^i = \left[\frac{\sum_{k=1}^n u_{ik} \sum_{j=1}^p (x_{kj} - v_j^i)^2}{\sum_{k=1}^n u_{ik}} \right]^{1/2} \quad (6.7)$$

Böylece, verilen girdi ve çıktı verilerden bulanık model oluşturulduktan sonra yapının tanımlanması aşaması tanımlanır.

6.1.2 Model parametrelerin belirlenmesi

Bu aşamada, kuralın soncul (THEN) tarafındaki koefsiyanlar (parametreler) belirlenir. Koefsiyanları belirlemek için lineer olmayan optimizasyon metodu kullanılır. Parametrelerin tahmini, lineer sistemlerin eşitliklerinin çözümüne dayanır.

Herhangi bir \tilde{y}_k bulanık çıktıya karşılık gelen $x_k=(x_{k1},x_{k2},\dots,x_{kp})$ için, bulanık çıkarım (5.3)'e benzer olarak aşağıdaki gibi olacaktır.

$$\tilde{y}_k = \frac{\sum_{i=1}^c A^i(x_k) \cdot y^i}{\sum A^i(x_k)} \quad (6.8)$$

Üyelik fonksiyonları ise aşağıdaki gibi normalleştirilir.

$$g_k^i = \frac{A^i(x_k)}{\sum_{i=1}^c A^i(x_k)} \quad (6.9)$$

Buradan \tilde{y}_k aşağıdaki gibi yazılabilir.

$$\tilde{y}_k = \sum_{i=1}^c g_k^i y^i = \sum_{i=1}^c g_k^i (b_0^i + \sum_{j=1}^p b_j^i x_{kj}) \quad (6.10)$$

Buradan açıkça görülmektedir ki; g_k^i bulanık kuralın öncül kısmından elde edilebilir. Ayrıca, bulanık modelin çıktısı \tilde{y}_k soncul kısımdaki $\{b_0^i, b_1^i, \dots, b_p^i\}$ parametrelerine de bağlıdır. Bu durumda, $\tilde{y}_k = x_{k(p+1)}$ eşitliği altında k tane lineer eşitlik aşağıdaki gibi tanımlanır:

$$\sum_{i=1}^c g_k^i (b_0^i + \sum_{j=1}^p b_j^i x_{kj}) = x_{k(p+1)} \quad (6.11)$$

Bu eşitliği vektör şeklinde gösterecek olursak;

$$Y = WB \quad (6.12)$$

Burada $Y=[x_{1(p+1)}, x_{2(p+1)}, \dots, x_{n(p+1)}]^T$, $B=[b_0^1, b_1^1, \dots, b_p^1, \dots, b_0^c, b_1^c, \dots, b_p^c]^T$ ve

$$W = \begin{bmatrix} g_1^1, g_1^1 x_{11} \dots g_1^1 x_{1p} \dots g_1^c, g_1^c x_{11} \dots g_1^c x_{1p} \\ g_2^1, g_2^1 x_{21} \dots g_2^1 x_{2p} \dots g_2^c, g_2^c x_{21} \dots g_2^c x_{2p} \\ \vdots \\ \vdots \\ g_n^1, g_n^1 x_{n1} \dots g_n^1 x_{np} \dots g_n^c, g_n^c x_{n1} \dots g_n^c x_{np} \end{bmatrix} \quad (6.13)$$

W matrisinin tersi kullanılarak, $\{b_o^i, b_1^i, \dots, b_p^i\}$ soncul parametreleri ařađıdaki gibi hesaplanabilir.

$$B = (W^T W)^{-1} W^T Y \quad (6.14)$$

Kuralın soncul kısımdaki parametrelerde belirlendikten sonra tahmin iřlemi (6.10) eřitliđi kullanılarak yapılır (Eminov vd, 2004).

6.2 Geliřtirilmiř Anahtarlamalı Bulanık Regresyon Modeli

Regresyon analizi bađımlı ve bađımsız deđiřken arasındaki iliřkiyi aıklayacak en uygun modelin bulunmasını sađlayan istatistiksel bir aratır. İstatistiksel yntemlerde, bu deđerler arasındaki iliřki aıklamak iin $f(x; \beta) + \varepsilon$ řeklinde tek fonksiyonel iliřki kullanılmaktadır. Bu tr tek fonksiyonel iliřkili modele alternatif olarak, bađımlı ve bađımsız deđiřken arasındaki iliřki $f_i(x; \beta_i) + \varepsilon_i$ řeklindeki c sayıda model ile de tahmin edilebilir. Fakat bu durumda veri setinin c sayıda alt veri setine ayrılması gerekmektedir. Veri setinin c sayıda alt sete ayrılması iin kmeleme analizi yaklařımı kullanılabilir. Kmeleme analizi kullanılarak veri seti Sert-C Ortalamalar veya Bulanık C-Ortalamalar yntemleri ile c sayıda alt sete ayrılabilir. Ama her iki ynteminde uygulanması sonucunda, regresyon modellerinin bulunması iin iki ařama gerekmektedir.

1. Ařama: Veri setinin Blnmesi ve yelik deđerleri matrisinin bulunması.
2. Ařama: Ađrılıklarılandırılmıř En Kk Kareler Yntemi kullanılarak regresyon modellerinin bulunması.

Ancak veri setinin bulunması, yelik deđerlerinin bulunması ve regresyon modellerinin bulunması tek ařamada gerekleřebilmektedir. Byle tahminler BCR Model ve kural tabanlı bulanık modellerle yapılabilir. BCR modeli, BCO algoritmasının geliřtirilmiř bir versiyonu olup veri setini genel anlamda, $f_i(x; b_i)$

biçimli kümelere ayrışımını gerçekleştiren bulanık kümeleme algoritmasıdır. BCR modeli kullanılarak S veri seti ayrıldıktan ve parametreler belirlendikten sonraki aşamada veri noktaları için hangi regresyon modelinin seçileceğidir. Bunun için ayrışım matrisi kullanılmaktadır. Herhangi bir veri noktası için uygun regresyon modeli seçerken, bu veri noktasının maksimum üyeliğe sahip olduğu küme aranır ve o kümeyle karşılık gelen regresyon modeli seçilir. Bu algoritmanın gerçekleştirilebilmesi için küme sayısı ve başlangıç ayrışım matrisinin ne olacağına karar verilmesi gerekmektedir. Aksi takdirde, belirlenen bir aralıktaki tüm küme sayıları için algoritmanın defalarca çalıştırılması gerekebilir. Bundan başka, başlangıçta ayrışım matrisinin rasgele üretilmesi, farklı sonuçlar elde edilmesine yol açabilir. Bu tür problemleri ortadan kaldırmak amacıyla geliştirilmiş Anahtarlamalı Bulanık Regresyon model algoritması aşağıdaki gibi tanımlanabilir (Eminov vd, 2005).

Geliştirilmiş Anahtarlamalı Bulanık Regresyon Model

Adım 1: Başlangıç

- Regresyon modelleri doğrusal fonksiyon olarak belirle,
- Uygun küme sayısını V_{sv} doğruluk indeksini kullanarak belirle ,
- Başlangıç ayrışım matrisi olarak BCO algoritmasının sonucunda elde edilen ayrışım matrisini kullan,
- m bulanık indeksini seç,
- ε işlem bitirme kriterini seç,

Adım 2: Parametrelerin Belirlenmesi: (5.8) eşitliğine göre,

$$J(U, \{\theta_i\}) = \sum_{i=1}^c \sum_{k=1}^N \mu_{ik}^m d_{ik}(\theta_i)$$

Amaç fonksiyonunu minimize edecek θ_i parametrelerini belirle.

Adım 3: Ayrışım Matrisinin Güncellenmesi: (5.10) eşitliğine göre,

$$\mu_{ik} = \frac{1}{\sum_{j=1}^c (d_{ik}/d_{jk})^{2/(m-1)}} \quad 1 \leq i \leq c, \quad 1 \leq k \leq N$$

Adım 4: İşlem bitirme kriterinin kontrol edilmesi:

$\|U^l - U^{l-1}\| \leq \varepsilon$ ise algoritmayı durdur. Aksi takdirde Adım2 'ye geri dön.

Adım 5: Anahtarlama: Uygun Modelin Seçimi.

$$y_i : x_k / u_{ik} > u_{jk}, i \neq j, i = 1, \dots, c, k = 1, \dots, n \quad (6.15)$$

Adım 6: Uzun Dönemli Tahmin (Forecasting): Anahtarlama Bulanık Regresyon modeli kullanılarak, uzun dönemli tahmin işlemi yapmak için ilk önce tahmini yapılacak veriye karşılık gelen bağımsız değişken için, c sayıda regresyon modelinden hangisinin kullanılacağına karar vermek gerekmektedir. Bu seçim için ilk önce bağımsız değişkene ilişkin tüm regresyon modelleri hesaplanır. Daha sonra ki adımda bu model değerleri ile kümeler arasındaki uzaklık hesaplanır ve hangi kümeyle olan uzaklık en küçük ise bu değişken için o regresyon modeli kullanılır. Bu regresyon modeli kullanılarak hesaplanan tahmin değeri bir sonraki adımda, bir sonraki dönemi tahmin edebilmek için bağımsız değişken olarak kullanılır.

7. GELİŞTİRİLMİŞ BULANIK MODELLER İLE TÜRKİYE’NİN ELEKTRİK TÜKETİMİNİN TAHMİNİ.

Bu tez çalışmasında, Bulanık C-Regresyon (BCRM) ve Takagi-Sugeno (T-S) bulanık modelleme teknikleri kullanılarak Türkiye’nin elektrik tüketimin tahmini yapılmaya çalışılmıştır. Bu amaca yönelik olarak veri setine, genellikle doğrusal olmayan fonksiyonel ilişkilerin ve karmaşık sistemlerin tahminini yapmak için kullanılan bulanık kümeleme analizine dayanan anahtarlamalı (switching) doğrusal regresyon modeli ve T-S modeli uygulanmıştır. Bunun için, Devlet İstatistik Enstitüsü (DİE)’nden elde edilen Türkiye’ye ilişkin 1970-2002 yıllarına arasındaki elektrik tüketim miktarları kullanılmıştır (Eminov, 2005).

Elektrik tüketim miktarlarına ilişkin değerler Ek Tablo1’de verilmiştir. Geliştirilen her iki bulanık model için, modelin yapısının ve ilgili parametrelerinin belirlenmesi için 1970-2002 yıllarına ait geçmiş elektrik tüketim verileri, eğitim ve test olmak üzere iki gruba ayrılmıştır. Bu doğrultuda, 1970-1990 yıllarına ilişkin veriler model belirlemek için başka bir deyişle eğitim seti olarak, 1991-1998 yıllarına ilişkin veriler 1.test verileri, 1999-2002 yıllarına ilişkin veriler ise 2. test verileri olarak kullanılmıştır (Hamzaçebi, 2004). Ayrıca söz konusu geliştirilmiş bulanık modellerin uygulanması sonucunda elde edilen modellerin uygunluğu test verileri ile kanıtlandıktan sonra, 2003-2010 yıllarına ilişkin uzun dönemli tahmin işlemi gerçekleştirilmiştir.

Tablo1’deki değerler kullanılarak ilk önce bu veri setine ilişkin en uygun küme sayısının bulunması gerekmektedir (Not: Bu değerler normalize edilmiştir ve normalize edilmiş değerler Ek Tablo2’de verilmiştir).

7.1 Geliştirilmiş Anahtarlamalı Bulanık Regresyon Modelin Uygulanması

Geliştirilmiş Anahtarlamalı Bulanık Regresyon modelin uygulanması için ilk önce modelde yer alacak değişkenler, modelin şekli, küme sayısı, başlangıç ayrışım matrisi, m bulanıklık indeksi, ε işlem bitirme kriteri gibi başlangıç değerlerinin belirlenmesi gerekmektedir. Başlangıç değerleri elde edildikten sonra model parametreleri belirlenerek tahmin işlemine geçilmektedir.

Bu uygulama için kullanılan model aşağıdaki gibidir.

$$y_i = b_{i0} + b_{i1} x \quad (7.1)$$

Burada

y = Elektrik Tüketim miktarını

x = Bir önceki yıla ilişkin tüketim miktarını göstermektedir.

7.1.1 Adım1: başlangıç değerlerinin girilmesi

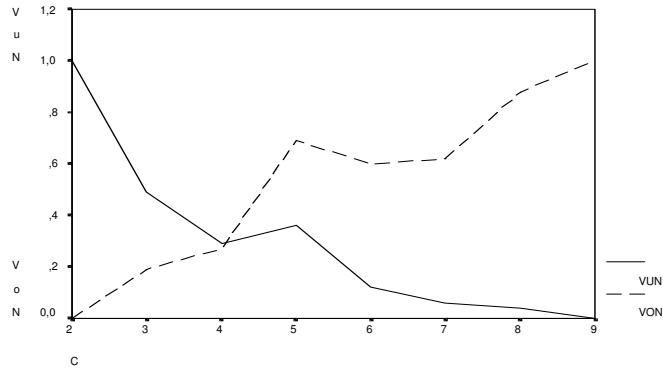
A) En uygun küme sayısının bulunması

Türkiye'nin 1970-1990 yılları arasındaki elektrik tüketim miktarlarına ($S=\{x_1, x_2, \dots, x_{20}\}$) göre en uygun küme sayısı bulunurken Bölüm 4'te değinilen V_{sv} doğrulama kriteri kullanılmıştır ve bu algoritmaya ilişkin program Borland Delphi 7 programla dilinde geliştirilmiştir ve Ek1'de verilmektedir.

Tablo 7.1 Farklı Küme Sayıları için Doğruluk İndeksi V_{sv}

Küme Sayısı	V_u	V_o	V_u^N	V_o^N	V_{sv}
2	0,001538	648,5789	1	0	1
3	0,000754	957,1757	0,48644	0,186882	0,673322
4	0,0004613	1099,0188	0,294707	0,27278	0,567487
5	0,0005584	1784,302	0,358313	0,687778	1,04609
6	0,0001982	1637,3582	0,122363	0,598791	0,721154
7	0,0000992	1674,8207	0,057513	0,621477	0,678991
8	0,0000691	2108,1568	0,037796	0,8839	0,921696
9	0,0000114	2299,8726	0	1	1

Program sonucunda, farklı küme sayılarına göre elde edilen V_{sv} değerleri Tablo 7.1'de görülmektedir. Bu doğrulama yöntemine göre en küçük V_{sv} değerini veren küme sayısı en uygun küme sayısı olarak seçilir. Tablo 7.1'den en uygun küme sayısının 4 olduğu görülmektedir.



Şekil 7.1 Farklı küme sayıları için V_u ve V_o Grafiği.

Aynı zamanda küme sayılarına göre V_u ve V_o değerlerinin grafiği çizildiğinde, bu değerlerin kesişim noktalarının da en uygun küme sayısını verdiği Şekil 7.1'den görülmektedir.

B) BCO algoritması kullanılarak başlangıç ayrışım matrisinin elde edilmesi

Verilere ilişkin uygun küme sayısı bulunduktan sonra BCRM algoritmasının uygulaması için gerekli olan diğer başlangıç değerleri elde edilmiştir. İlk adımda BCO kümeleme algoritması kullanılarak BCRM algoritmasının başlangıç üyelik değerleri matrisi elde edilmiştir. Bu değerler Tablo 7.2'de gösterilmektedir.

Tablo 7.2 BCO Algoritması Sonucunda Elde Edilen Üyelik Değerleri.

U1	U2	U3	U4
0,004677	0,040994	0,011794	0,942535
0,001718	0,017032	0,004492	0,976758
0,000152	0,001731	0,000412	0,997706
0,00027	0,003521	0,000761	0,995447
0,003561	0,059369	0,010684	0,926387
0,012361	0,351901	0,041676	0,594063
0,011089	0,742551	0,042852	0,203508
0,005147	0,915163	0,022043	0,057648
0,002037	0,971106	0,00935	0,017506
0,000394	0,994947	0,001932	0,002727
0,001037	0,98801	0,005787	0,005165
0,009795	0,891054	0,066863	0,032288
0,021221	0,752786	0,174011	0,051982
0,03712	0,424536	0,482454	0,05589
0,023523	0,095794	0,861209	0,019474
0,001118	0,002067	0,996272	0,000543
0,074744	0,043903	0,866309	0,015045
0,437518	0,06878	0,465563	0,02814
0,837505	0,027897	0,12181	0,012788
0,990836	0,001998	0,006135	0,001031
0,916288	0,020309	0,052186	0,011217

BCO kümeleme algoritması, veri setine uygulandıktan sonra elde edilen kümeler Tablo 7.3'te görüldüğü gibidir.

Tablo 7.3 BCO Algoritması Uygulandıktan Sonra Elde Edilen Kümeler.

1.Küme		2.Küme		3.Küme		4.Küme	
X	Y	X	Y	X	Y	X	Y
39.721,5	43.120,0	16.068,90	17.968,80	24.465,10	27.635,20	7.307,80	8.289,30
43.120,0	48.633,60	17.968,80	18.933,80	27.635,20	29.708,60	8.289,30	9.527,30
48.633,6	50.295,70	18.933,80	19.663,10	29.708,60	32.209,70	9.527,30	10.530,10
		19.663,10	20.398,20	32.209,70	36.697,30	10.530,10	11.358,70
		20.398,20	22.030,00	36.697,30	39.721,50	11.358,70	13.491,70
		22.030,00	23.586,80			13.491,70	16.068,90
		23.586,80	24.465,10				

Kümeleme işlemi, 4. Bölümde değinildiği gibi gözlemin maksimum üyeliğine göre yapılır. Her bir gözlem maksimum üyeliğe sahip olduğu kümeye tahsis edilir. BCO algoritması sonucunda elde edilen kümelere ilişkin küme prototipleri $c=4$ için Tablo 7.4'te verilmektedir. Kümelere giren gözlem değerleri bu küme merkezleri etrafında küresel bir biçimde dağılmaktadır. Ayrıca bu veri seti için elde edilen amaç fonksiyonu değeri ise 0.00816'dır.

Tablo 7.4 Küme Merkezleri (Centroid of Clusters).

V1	44631,44	45673,162
V2	20827,34	20096,254
V3	30212,364	32884,708
V4	10048,183	10651,54

Bu kümeleme algoritmasına ilişkin kaynak program Borland Delphi 7 programlama dilinde geliştirilmiştir ve Ek 2'de verilmiştir. Yapılan denemeler sonucunda bulanıklık indeksi $m=2$ ve işlem bitirme kriteri $\varepsilon < 0.0000000001$ olarak belirlenmiştir.

7.1.2 Adım 2: model parametrelerin belirlenmesi.

Bu bölümde Türkiye'nin elektrik tüketimin modellenmesi için BCRM algoritması kullanılmıştır. (7.1) modelleri, tahmin algoritmasının eğitim setine ilişkin verilere uygulanması ile elde edilir. Bu tahmin algoritması için tüm başlangıç

değerleri elde edildikten sonra t iterasyon adımında (6.15) amaç fonksiyonu minimize edilerek $U^{(t)}$ ayrışım matrisine karşılık gelen $\{b_0^i, b_1^i\}$ parametreleri elde edilmiştir. Elde edilen sonuçlara göre regresyon modelleri de aşağıdaki gibi bulunmuştur.

$$y_1 = 0.0043 + 1.1107x$$

$$y_2 = 0.0009499 + 1.0451x \quad (7.2)$$

$$y_3 = -0.0006947 + 1.003x$$

$$y_4 = 0.0001298 + 1.0767x$$

Herhangi bir yıla karşılık gelen x_k tüketim değerinin tahmini için kullanılacak en uygun model, (x_k, y_k) veri noktası için kümelere olan üyeliklerine göre seçilecektir. Bir başka deyişle bu veri noktası için en uygun model maksimum üyeliğe sahip olduğu kümenin regresyon modeli olacaktır. Bu modelleme yöntemi Bölüm 6'da değinilen geliştirilmiş BCRM'dir (Eminov vd , 2005).

7.1.3 Adım3: ayrışım matrisinin güncellenmesi

Ayrışım matrisi işlem bitirme kriteri sağlanana kadar her t iterasyonunda (6.16) göre güncellenmelidir.

Tablo 7.5 BCRM Algoritmasında sonucunda elde edilen üyelik değerleri

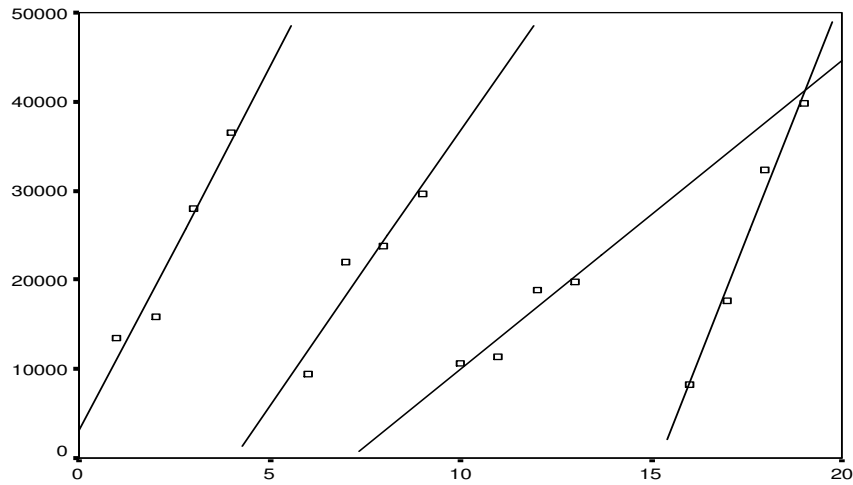
U1	U2	U3	U4
0,000883	0,017714	0,033121	0,948282
0,00484	0,912927	0,012535	0,069697
0,010248	0,173805	0,427391	0,388556
0,000239	0,001861	0,995576	0,002324
0,992644	0,003124	0,001265	0,002966
0,947015	0,020832	0,00961	0,022544
0,075319	0,281386	0,039704	0,603591
0,004517	0,044069	0,925566	0,025849
0,001958	0,012603	0,977313	0,008126
0,001586	0,010317	0,981645	0,006453
0,000686	0,98024	0,00236	0,016715
0,008806	0,832196	0,049495	0,109503
0,000516	0,00407	0,993324	0,00209
0,787262	0,064078	0,020731	0,12793
0,003713	0,901977	0,007632	0,086678
0,003308	0,039681	0,002947	0,954064
0,990867	0,002781	0,001129	0,005222
0,001366	0,008894	0,000896	0,988845
0,002746	0,009069	0,001232	0,986953
0,999976	0,000007	0,000003	0,000015

Tablo 7.5'deki üyelik değerlerine göre veri setine bu algoritmanın uygulanması sonucunda elde edilen kümeler Tablo 7.6'da görülmektedir.

Tablo 7.6 BCRM algoritması sonucunda elde edilen kümeler.

1.Küme		2.Küme		3.Küme		4.Küme	
X	Y	X	Y	X	Y	X	Y
11358,70	13491,7	8289,30	9527,30	9527,30	10530,1	7307,80	8289,30
13491,70	16068,9	20398,2	22030,0	10530,1	11358,7	16068,9	17968,8
24465,10	27635,2	22030,0	23586,8	17968,8	18933,8	29708,6	32209,7
32209,70	36697,3	27635,20	29708,6	18933,80	19663,10	36697,30	39721,50
43120,00	48633,60			19663,10	20398,20	39721,50	43120,00
				23586,80	24465,10		

Tablo 7.6'dan görüldüğü gibi tahmin işlemi yapılırken 1. küme için 1. regresyon modeli, 2. küme için 2. regresyon modeli vs. kullanılmıştır.



Şekil 7.2 Kümelere göre tahmin değerleri.

Şekil 7.2'den eğitim seti için kümelere göre gerçek ve tahmin değerleri ile elde edilen regresyon modelleri görülmektedir.

7.1.4 Adım 4: tahmin

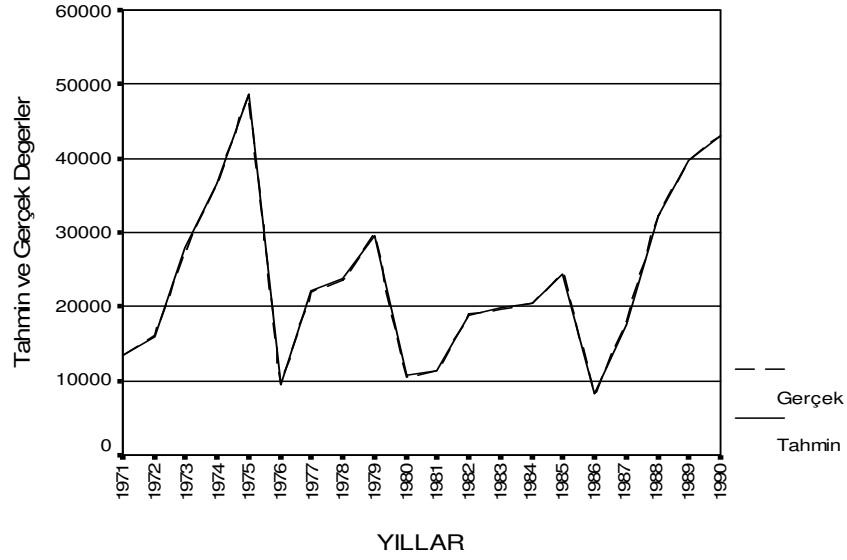
Bu modelleme yöntemin doğruluğunu kanıtlamak amacıyla daha önce yapılan çalışmalarla (Hamzaçebi, 2004) karşılaştırma yapılmıştır ve karşılaştırma kriteri olarak da Mutlak Yüzde Hata (MYH) ve Ortalama Mutlak Yüzde (OMYH)

kullanılmıştır (Hamzaçebi, 2004). Eğitim seti için BCRM algoritması ile elde edilen tahmin değerleri, MYH'lar ve OMYH Tablo 7.7'de görülmektedir. Ayrıca elde edilen bu modeller kullanılarak elektrik tüketim miktarlarının 2003-2010 yıllarına ait uzun dönemli tahmini yapılmıştır.

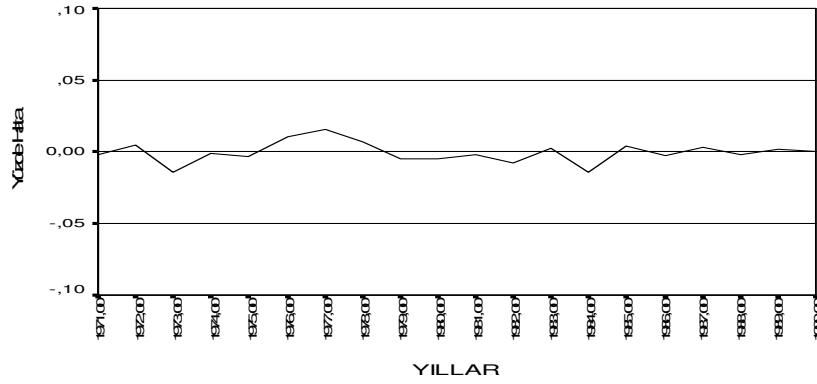
Tablo 7.7 1970-1990 Yıllarına İlişkin Tahmin ve MYH Değerleri.

GERÇEK	TAHMİN	MYH(%)
8.289,30	8.303,23	0,168
9.527,30	9.485,63	0,437
10.530,10	10.680,96	1,433
11.358,70	11.372,80	0,124
13.491,70	13.539,56	0,355
16.068,90	15.896,07	1,076
17.968,80	17.688,93	1,558
18.933,80	18.799,14	0,711
19.663,10	19.762,54	0,506
20.398,20	20.490,63	0,453
22.030,00	22.072,08	0,191
23.586,80	23.768,23	0,769
24.465,10	24.407,81	0,234
27.635,20	28.019,30	1,390
29.708,60	29.594,49	0,384
32.209,70	32.301,03	0,284
36.697,30	36.575,41	0,332
39.721,50	39.787,97	0,167
43.120,00	43.027,77	0,214
48.633,60	48.628,93	0,010
	OMYH:	0,540

Tablo 7.7'den tahmin ve gerçek değerler arasındaki yüzdelik farkın maksimum 1.56 olduğu görülmektedir. Buradan BCRM yönteminin iyi bir tahminleme aracı olarak kullanılabilirliği söylenebilir.



Şekil 7.3 BCRM yönteminin eğitim seti için performansı.



Şekil 7.4 BCRM yöntemi için modellenmenin yüzde hatası.

7.2 Geliştirilmiş Takagi-Sugeno Modelleme Algoritmasının Uygulanması

Ayrıca, bu çalışmada Türkiye'nin elektrik tüketiminin uzun dönemli tahmini için 6.Bölümde değinilen bulanık modelleme yöntemlerinden Takagi-Sugeno (T-S) modelleme yöntemi de kullanılmıştır. Bu yöntem için $m=1.5$ değerinde en uygun küme sayısı 2 olarak bulunmuştur. Bu yöntemle modelleme işlemi yapılırken BCRM yönteminde olduğu gibi veri seti eğitim seti (1970-1990), 1. test verileri (1991-1998) ve 2. test verileri olmak (1999-2002) üzere 3'e ayrılmıştır. Eğitim seti için bulunan küme merkezleri, standart sapma ve üyelik değerleri sırasıyla Tablo 7.8, Tablo 7.9'da görülmektedir.

Tablo 7.8 T-S Modeli Standart Sapma ve Küme Merkezi Değerleri.

R^1	ρ_{11}	ρ_{12}
	0,004301	0,04692
	δ_1	
	0,002983	
R^2	ρ_{21}	ρ_{22}
	0,014245	0,162243
	δ_2	
	0,003338	

Burada R^1 ve R^2 bulanık kuralları, ρ_{ij} 'ler küme merkezlerini, δ_i 'ler ise standart sapmaları göstermektedir.

Tablo 7.9 T-S modelleme yöntem için elde edilen üyelik değerleri.

U1	U2
0,353819	0,000111
0,447754	0,000212
0,577174	0,000467
0,684419	0,000858
0,769387	0,001393
0,941784	0,004476
0,993961	0,015758
0,905113	0,035824
0,826455	0,052516
0,756769	0,069043
0,680883	0,089748
0,506871	0,153063
0,353839	0,239357
0,279369	0,299886
0,097354	0,576124
0,041197	0,770581
0,012202	0,948239
0,000841	0,928798
0,000097	0,68904

Bu modelleme yönteminden elde edilen kurallar ise aşağıdaki gibidir.

$$R^1 : \text{If } \bar{X} \text{ is } A^1(\rho_{11}, \rho_{12}, \delta^1) \text{ then } y^1 = -0.005 + 11.745x \quad (7.3)$$

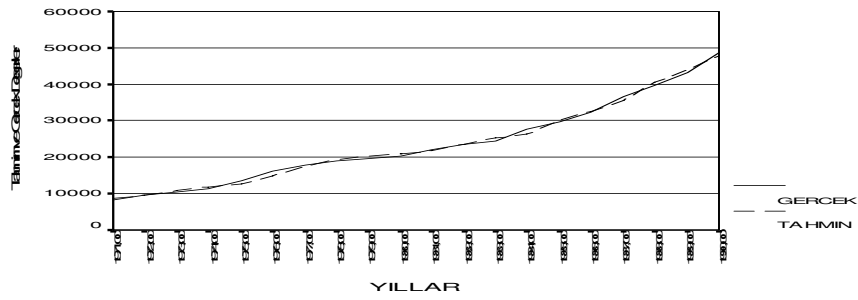
$$R^2 : \text{If } \bar{X} \text{ is } A^2(\rho_{21}, \rho_{22}, \delta^2) \text{ then } y^1 = 0.00202 + 10.4174x$$

Bu T-S bulanık modellerine dayanarak elde edilen tahmin değerleri, gerçek değerler, MYH'lar ve OMYH Tablo 7.10'da görülmektedir.

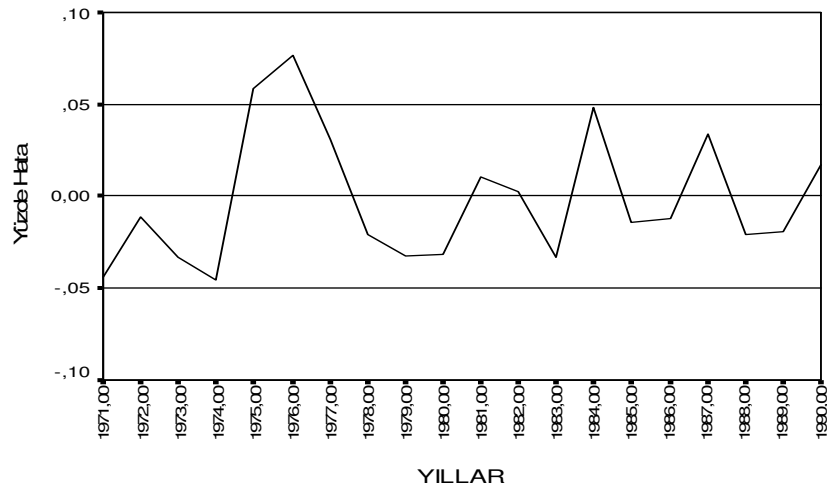
Tablo 7.10 T-S Bulanık Modelleme yöntemine göre elde edilen tahmin, gerçek ve MYH değerleri.

Gerçek	Tahmin	MYH
8289,30	8655,34	4,416
9527,30	9637,1	1,152
10530,10	10875,03	3,276
11.358,70	11877,71	4,57
13491,70	12706,15	5,822
16068,90	14838,61	7,656
17968,80	17416,34	3,075
18933,80	19322,87	2,055
19663,10	20297,84	3,228
20398,20	21040,68	3,150
22030,00	21797,64	1,055
23586,80	23526,8	0,254
24465,10	25272,1	3,299
27635,20	26303,1	4,820
29708,60	30136,36	1,440
32209,70	32586,36	1,169
36697,30	35462,89	3,364
39721,50	40543,79	2,070
43120,00	43954,69	1,936
48633,60	47786,04	1,743
	OMYH:	2,977

Buradan BCRM algoritmasının T-S bulanık modelleme yönteminden daha iyi sonuç verdiği görülmektedir.



Şekil 7.5 T-S yönteminin eğitim seti için performansı.



Şekil 7.6 T-S modelleme yöntemi için yüzde hatalar.

T-S modelleme yöntemine ilişkin program Borland Delphi 7 programlama dilinde geliştirilmiş ve Ek 4'te verilmiştir.

7.3 Tahmin Modellerinin Değerlendirilmesi

Geliştirilmiş bulanık modellerin (BCRM ve Takagi-Sugeno) iyi bir tahminleme aracı olduğu kanıtlamak amacıyla, test verileri için (Hamzaçebi, 2004), çalışmada kullanılan Box-Jenkins, Regresyon, Yapay Sinir Ağları gibi zaman serilerini tahminleme amacıyla kullanılabilen teknikler ile karşılaştırılması yapılmıştır.

1. test verileri için elde edilen tahmin ve gerçek değerleri, Tablo 7.11'de görülmektedir.

Tablo 7.11 1991-1998 Yıllarına İlişkin Tahmin Değerleri.

TAHMİN(GWH)										
YIL	GERÇEK (GWh)	ARIMA (0,1,1)	ARIMA (1,1,0)	ARIMA (2,2,0)	ARIMA (0,2,1)	RT	BPN	RBFN	TS	BCRM
1991	50295,7	52864,2	53459,4	53102,8	53247,1	49830,42	50495	50068	54001,34	49398,0132
1992	54613,1	55014,8	57748,3	57295,8	58046,9	53248,99	54477	54491	55874,96	54352,4726
1993	60406,3	57165,4	61617,9	62685,1	63033,1	56802,73	58663	59237	60741,79	61323,0150
1994	61420,3	59316	65160,2	67954,7	68205,7	60491,65	63045	64277	67272,22	61144,8328
1995	67092,3	61466,6	68447,1	72971,5	73564,5	64315,76	67609	69574	68415,26	66266,7999
1996	74326,8	63617,2	71534,4	78610,3	79109,7	68275,05	72342	75077	74809,08	75109,4342
1997	81884,9	65767,8	74465,9	84446,8	84841,3	72369,51	77226	80728	82964,23	83101,7613
1998	87704,6	67918,4	77275,8	90195,6	90759,2	76599,16	82239	86459	91484,17	88184,1671

1991-1998 yıllarına ilişkin bütün yöntemlere göre MYH ve OMYH değerleri ise Tablo 7.12'de görülmektedir.

Tablo 7.12 1991-1998 Yıllarına İlişkin MYH(%) ve OMYH

MYH(%)									
YIL	ARIMA (1,1,0)	ARIMA (1,1,0)	ARIMA (2,2,0)	ARIMA (0,2,1)	RT	BFN	RBFN	TS	BCRM
1991	5,1068	6,2902	5,5812	5,8681	0,9250	0,3962	0,4527	7,37	1,7848
1992	0,7355	5,7407	4,9122	6,2875	2,4977	0,2492	0,2235	2,31	0,4772
1993	5,3652	2,0058	3,7725	4,3486	5,9655	2,8859	1,9357	0,56	1,5175
1994	3,4261	6,089	10,639	11,047	1,5119	2,6452	4,6510	9,53	0,4484
1995	8,385	2,0193	8,7629	9,6467	4,1383	0,7701	3,6989	1,97	1,2303
1996	14,409	3,7569	5,7631	6,435	8,1420	2,6703	1,0093	0,65	1,0529
1997	19,683	9,063	3,1287	3,6104	11,6204	5,6895	1,4128	1,32	1,4860
1998	22,56	11,891	2,8402	3,4828	12,6623	6,2318	1,4202	4,31	0,5467
OMYH:	9,9588	5,8566	5,6749	6,3408	5,9329	2,6923	1,8505	3,50	1,0680

2. test verileri (1999-2002) için ise tahmin ve gerçek değerler Tablo 7.13'te görülmektedir.

Tablo 7.13 1999-2002 Yıllarına İlişkin Tahmin Değerleri.

TAHMİN (GWh)										
YIL	Gerçek	ARIMA (0,1,1)	ARIMA (1,1,0)	ARIMA (2,2,0)	ARIMA (0,2,1)	RT	BFN	RBFN	TS	BCRM
1999	91201,9	70069	79991	96279	96863	90781,1	87358	92197	98044,47	92035,12
2000	98295,7	72220	82632	102625	103154	96413,0	92558	97863	101986,84	98162,57
2001	97070	74370	85215	109021	109631	102237,1	87811	103370	109983,38	98950,93
2002	102800	76521	87753	115631	116294	108253,7	103090	108650	108601,70	101770,54

Bu verilere ilişkin MYH ve OMYH değerleri ise Tablo 7.14'te görülmektedir.

Tablo 7.14 1999-2002 Yıllarına İlişkin MYH (%) ve OMYH.

MYH (%)									
YIL	ARIMA (0,1,1)	ARIMA (1,1,0)	ARIMA (2,2,0)	ARIMA (0,2,1)	RT	BFN	RBFN	TS	BCRM
1999	23,1715	12,2923	5,5668	6,2072	0,4613	4,2147	1,0910	7,50	0,9136
2000	26,5278	15,9352	4,4043	4,9425	1,9153	5,8371	0,4402	3,76	0,1354
2001	23,3851	12,2128	12,3117	12,9425	5,3231	0,7633	6,4901	13,30	1,9377
2002	25,5632	14,6371	12,4815	13,1264	5,3051	0,2821	5,6906	5,64	1,0014
OMYH:	24,6619	13,7694	8,6911	9,3040	3,2515	2,7743	3,4280	7,55	0,9970

Yukarıdaki tablolardan da anlaşılacağı üzere tanımlanan BCRM tabanlı regresyon modeli ile tüm diğer yöntemlerden daha doğru tahmin yapılmıştır.

7.4 Uzun Dönemli Tahmin

Son olarak, belirlenen modelle 2003-2010 yıllarına ait uzun dönemli elektrik tüketim tahmini yapılmış, onun ve söz konusu modellerin tahmin sonuçları karşılaştırmalı olarak Tablo 7.15'de verilmiştir.

Tablo 7.15 2003-2010 Uzun Dönemli Tahmin Sonuçları.

YIL	ARIMA (0,1,1)	ARIMA (1,1,0)	ARIMA (2,2,0)	ARIMA (0,2,1)	RT	BFN	RBFN	BCRM	Gerçek
2003	78671	90255	122492	123144	114462,63	108360	113610	110079,57	111766
2004	80822	92730	129481	130180	120863,91	113600	118180	117918,02	121142
2005	82973	95184	136648	137402	127457,55	118780	122290	126358,27	129573
2006	85123	97620	144038	144811	134243,55	123870	125880	135446,51	
2007	87274	100044	151595	152406	141221,91	128850	128890	145232,49	
2008	89424	102457	159324	160187	148392,64	133690	131300	151584,54	
2009	91575	104863	167255	168155	155755,74	138380	133070	158223,21	
2010	93726	107262	175369	176309	163311,6	142890	134180	165161,45	

8. TARTIŞMA VE SONUÇLAR

Bu çalışmada, veri noktalarını aralarındaki benzerlikler kullanılarak kendi içinde homojen kendi aralarında heterojen gruplara ayırmayı sağlayan kümeleme analizi yaklaşımları ve bu yaklaşımlardan bulanık kümeleme yaklaşımının bulanık modellemeye uygulamaları detaylı bir şekilde incelenmiştir.

Genelde modelleme ve tahmin işlemi yapılırken, zaman serileri ve regresyon gibi geleneksel istatistiksel teknikler kullanılmaktadır. Fakat gerek regresyon modelinin gerekse Box-Jenkins modellerinin doğrusal olmayan sistem davranışlarını modellemede pek başarılı sonuçlar vermediği daha önce yapılan çalışmalardan görülmektedir. Ayrıca bu modeller veri sayısının azlığı, hatanın normal dağılımına uymama, bağımsız ve bağımlı değişken arasındaki ilişkinin belirsizliği gibi durumlarda yetersiz kalır. Bu nedenle doğrusal olmayan ve karmaşık sistemleri daha doğru bir şekilde tanımlamak ve modelleyebilmek için son zamanlarda Bulanık mantık, Yapay Sinir Ağları (YSA), Genetik Algoritma gibi esnek teknikler (Soft Computing) uygulanmaktadır. Bu esnek teknikler içerisinde yaygın olarak kullanılan YSA modelinde ağ yapısının bulunması ve gizli katmandaki nöron sayısının belirlenmesi gibi bir takım zorluklar bulunmaktadır.

Bu çalışmada, bulanık kümelemeye dayanan Mamdani, Takagi-Sugeno, Anahtarlama Bulanık Regresyon, Sugeno-Yasukawa gibi yaygın kullanılan bulanık modeller incelenmiştir. Bu tür modelin kurulmasında küme sayısı, başlangıç ayrışım matrisi vs. gibi bir takım başlangıç şartlarının sağlanması gerekmektedir. Daha önce yapılan çalışmalarda bu tür başlangıç koşulları modelleme sürecinin değişik başlangıç koşullarında tekrarlanması sonucu elde edilmekteydi. Sistemin modellenmesi için, modelleme sürecinin bu şekilde çok sayıda tekrarlanması gerekliliği bu modelleme yöntemlerinin dezavantajı olarak ortaya çıkmaktadır. Bu çalışmada optimal küme (kural) sayısının ve başlangıç ayrışım matrisinin elde edilmesi için kümeleme algoritmalarından Bulanık C-Ortalama algoritması ve V_{sv} doğruluk indeksine dayanan algoritmanın kullanılması önerilmiştir. Bu doğrultuda, yukarıda bahsedilen bulanık modelleme yöntemlerinden Takagi-Sugeno ve Anahtarlama Bulanık Regresyon modellerinin iyileştirilmesi amaçlanmıştır.

Takagi-Sugeno modelleme birbirinden bağımsız yürütülen yapısal tanımlama ve parametrik tanımlama olmak üzere iki aşamayı içerir. Bu tezde Takagi- Sugeno bulanık modeli iyileştirilir ve V_{sv} doğruluk indeksi kullanılarak en uygun küme (kural sayısının) sayısı ve Bulanık C-Ortalamalar Algoritması kullanılarak başlangıç ayrışım matrisi önceden belirlenir. Kuralın öncül kısmı tek sayıda bulanık kümeye uygulanır ve ona karşılık gelen çoğul değişkenli Gaussian biçimli üyelik fonksiyonunun belirlenmesi için, ortalamalar ve tek standart sapma gibi parametrelerin belirlenmesi yeterlidir. Bulanık kuralın soncul kısmındaki doğrusal denklem parametreleri En Küçük Kareler yöntemiyle belirlenir. Böylece, bulanık kuralların önceden belirlenmesi ve az sayıda belirlenecek öncül parametre olmasından dolayı bu tür modelleri kurmak daha kolay ve hesaplanma açısından daha az maliyetlidir. Anahtarlamalı Bulanık Regresyon bulanık modelinin iyileştirilmesine yönelik ise ayrışım matrisinin başlangıç değerlerinin seçimi yine aynı şekilde yapılmaktadır. Ayrıca Anahtarlamalı Bulanık Regresyon modelinin iyileştirilmesinde bir başka önemli husus bu bulanık modeli kullanarak uzun dönemli tahmin yapılmasına olanak sağlamasıdır.

Söz konusu iyileştirilmiş bu iki bulanık model Türkiye'nin elektrik tüketiminin tahminine (uzun dönemli dahil) uygulanmıştır. Bu amaçla her iki bulanık modelin uygulanabilmesi için ilk önce veri setine uygun küme sayısı bulunmuştur. Bunun sonucunda, Takagi-Sugeno Modeli için $m=1.5$ değerinde en uygun küme sayısı başka bir deyişle kural sayısı 2 olarak belirlenmiştir. Anahtarlamalı Bulanık Regresyon modeli için ise $m=2$ değerinde en uygun küme sayısı 4 olarak bulunmuştur. Bulanık C-Ortalamalar ve Bulanık C-Regresyon kümeleme algoritmaları kullanılarak ayrışım matrisi bulunduktan sonra modellemenin diğer adımları gerçekleştirilmiştir. Bu iki bulanık model kullanılarak Türkiye'nin elektrik tüketim miktarlarının modelleme ve tahmini yapmak için DİE'den temin edilen 1970-2002 yıllarına ilişkin elektrik tüketim miktarları kullanılmıştır. 1970-1990 yılları arasındaki veriler model yapısının ve parametrelerinin belirlenmesi için kullanılan eğitim seti olarak, 1991-1998 ve 1999-2002 yılları arasındaki veriler ise sırasıyla 1. test verileri ve 2. test verileri olarak kullanılmıştır. Eğitim setine göre elde edilen modeller kullanılarak tahmin işlemi yapılmış ve tahmin sonuçları istatistiksel regresyon, Box-Jenkins ve YSA gibi diğer

modelleme tekniklerinden elde edilen sonuçlar ile karşılaştırılmıştır. Sonuçlara bakıldığında; 1991-1998 yıllarına ilişkin Anahtarlama Bulanık Regresyon için OMYH 1,0680 olarak elde edilmiş ve böylece diğer modelleme tekniklerinden daha iyi bir sonuç sağlanılmıştır. 1991-1998 yılları için Takagi-Sugeno Modelleme ile elde edilen OMYH değeri 3.5 olarak bulunmuş ve bu sonucun istatistiksel sonuçlardan daha iyi olduğu görülmektedir. Aynı şekilde her iki bulanık model için 1999-2002 yıllarına ilişkin tahmin işlemi yapılmış ve OMYH'lar Takagi-Sugeno için 7.55 ve Anahtarlama Bulanık Regresyon için ise 0.9970 olarak elde edilmiştir. Bu sonuçlardan da, Anahtarlama Bulanık Regresyon modelinin diğer tüm yöntemlerden, Takagi-Sugeno modelinin ise zaman serileri yönteminden daha iyi sonuç verdiği görülmektedir. Son olarak, Anahtarlama Bulanık Regresyon yöntemi kullanılarak Türkiye'nin 2003-2010 yılları arasındaki elektrik tüketimin uzun dönemli tahmin öngörüsü yapılmıştır ve 2003, 2004, 2005 yıllarına ait gerçek tüketim miktarlarına bakıldığında, bu modelleme yönteminin daha doğru tahminler elde ettiği anlaşılmıştır.

Daha sonraki çalışmalarda Anahtarlama Bulanık Regresyon yöntemi için, veri setinin yapısına bağlı olarak c sayıda doğrusal regresyon modeli kullanmak yerine, parabolik, kubik, üstel, kuvvet vs. regresyon modelleri kullanılabilir.

KAYNAKLAR

Abonyi, J.; Babuska, R.; Chovan, T.; Szeifert, F., 2000, Incorporating Prior Knowledge in Fuzzy C-Regression Models-Application to System Identification, The Netherlands.

Akyılmaz, O., Esnek Hesaplama Yöntemlerinin Jeodezide Uygulamaları, Doktora Tezi., İTÜ Fen Bilimleri Ens., İstanbul, 2005.

Babuska, R., Fuzzy Systems, Modeling and Identification, Delft University of Technology Department of Electrical Engineering.

Balasko, B.; Abonji, J.; Feil, B., Fuzzy Clustering and Data Analysis Toolbox. www.fmt.vein.hu/softcomp/fclusttoolbox/fuzzyclusteringtoolbox.pdf.

Berry, M.J.A.; Linoff, G., 1996, Data Mining Techniques For Marketing, Sales and Customer Support, John Wiley&Sons, Inc., USA.

Bezdek, J., 1981, *Pattern Recognition with Fuzzy Objective Functions*, Plenum Press, New York.

Bezdek, J.C.; Coroy, C.; Gunderson, R.; Watson, J., 1981, Detection and Characterization of Cluster Substructure, SIAM, Journ. Appl. Math. 40, 339-372.

Bezdek, J.C.; Ehrlich R.; Full, W., 1984, FCM: Fuzzy C-Means Algorithm Computers and Geoscience 10 (2-3), 191-203.

Bezdek, J.C.; Pal, S.K., 1992, Fuzzy Models For Pattern Recognition: Methods that Search for Structures in Data, IEEE Pres, New York,

Bock, H.H., 1979, *Cluster Analyse Mit Unscharfen Partitionen*, in :H.H.Bock Klassifikation un Erkenntnis: Vol:III: Numerische Klassifikation INDEKS, Frankfurt, 137-163.

Dave, R.N, 1989, Use of Adaptive Fuzzy Clustering Algorithm to Detect Lines in Digital Images, Proc. Intelligent Robots and Computer Vision VIII, Vol:1192, 600-611.

Dave, R.N., 1996, Validating Fuzzy Partitions Obtained Through c-Shells Clustering, Pattern Recognition Letters, 17, 613-623.

Duda, R.,; Hart, P., 1973, Pattern Classification and Scene Analysis, Wiley, New York.

Dunn, J.C., 1974, A Fuzzy Relative of ISODATA Process and Its Use in Detecting Compact, Well Separated Clusters, Journ., Cybern., 3, 95-104.

Eminov, M.,; Dilek, M.; Güler, N., 2005, Bulanık Doğrusal Modelleme ile Türkiye'nin Elektrik Enerjisi Tüketiminin Uzun Dönemli Tahmini , Bilimde Modern Yöntemler Sempozyumu, 16-18 Kasım,2005, Kocaeli.

Eminov, M.; Güler, N., 2004, Fuzzy C-Means Clustering Based Method for Fuzzy Linear Modeling, International Conference on Computational Intelligence, May 27-29, 2004, Nicosia, North Cyprus.

Gath, J.; Geva, A.B., 1989, Unsupervised Optimal Fuzzy Clustering, IEEE Transactions on Pattern Analysis and Machine Intelligence, 11(7), 773-781.

- Geraldo, X., Fuzzy Logic, Computing Science Department and Systems and Computing , Lecture Notes.
- Gustafson, E.E; Kessel, W.,C, 1979 , Fuzzy Clustering with a Fuzzy Covariance Matrix, IEEE CDC, San Diego, California, 761-245.
- Halkıdı, M.,; Batıstakıs, Y.,; Vazırđıannıs, M., 2001, On Clustering Validation Techniques, Journal of Intelligent Information Systems, Kluwer Academic Publishers , Manufactred in the Netherlands, 17:2/3 pp:107-145.
- Hamarat, B., *Türkiye’de Sağlık Açısından Homojen İl Gruplarının Belirlenmesine İlişkin İstatistiksel bir Yaklaşım*, Y.Lisans Tezi, Anadolu Üni. Fen Bilimleri Ens., Eskişehir, 1998.
- Hamzaçebi, C., ; Kutay F., 2004, Yapay Sınır Ağları ile Türkiye’nin Elektrik Enerjisi Tüketiminin 2010 Yılına Kadar Tahmini, Gazi Üniv.Müh.Mim.Fak. Dergisi, Cilt:19, No:3, 227-233.
- Han, J., Cluster Analysis, www.cs.sfu.ca/~han/bk/8clst.ppt.
- Han, J.,; Kamber, M., 2000, *Data Mining Concepts and Techniques* , Morgan Kaufman Pulishers, 1 st Ed., Francisco , USA.
- Hartigan, J. A., 1975, Clustering Algorithms, John Wiley & Sons, Inc, New York,
- Hathaway, R.,J.; Bezdek, J.C., 1993, Switching Regression Models and Fuzzy Clustering, IEEE Transactions on Fuzzy Systems, Vol.1, No.3.
- Hinneburg, A.; Keim, D., 1998, An Efficient Approach to Clustering in Large Multimedia Databases With Noise, In Proceedings of KDD Conference.
- Höppner, F.;Klawonn, F.;Rudolf, K.; Runkler, T., 1999, *Fuzzy Cluster Analysis*, Wiley.
- Jain, A.,; Dubes, R., 1998, Algorithms for Clustering Data, Prentice-Hall, Englewood, Cliffs, NU.
- Kaufman, L., ; Rousseuw, P., 1990, *Finding Groups Data : An Introduction to Cluster Analysis*, John Wiley and Sons, New York.
- Kim, E.; Park, M.; Ji, S.;Park M., 1997, A New Approach to Fuzzy Modeling , IEEE Trans., Fuzzy Systems, Vol:5, no:3, pp:328-337
- Koçyiğit, Y.; Korükek, M., 2005, EMG İşaretlerini Dalgacık Dönüşümü ve Bulanık Mantık Sınıflayıcı Kullanarak Sınıflama, İTÜ Dergisi/d Mühendislik, Cilt:4, Sayı:3, 25-31 Haziran 2005.
- Mamdani, E.H.; Assilian, S., 1975, An Experiment in Linguistic Synthesis with a Fuzzy Logic Controller, Int. Journal of Man-Machine Studies, 7(1),1-13.
- Michalski, R.,S.,; Step, R.,E.,; Diday, E., 1981, A Recent Advance in Data Analysis : Clustering Objects into Classes Characterized by Conjunctive Concepts, North Holland Publishing Company .
- Milligan, G.W.;Soon, S.C.; Sokol, L.M., 1983, The Effect of Cluster Size, Dimensionality and Number of Clusters on Recovery of True Cluster Structure, IEEE Transactions on Pattern Analysis and Machine Intelligence, 5, 40-47.

- Naes, T.; Mevik, B., H., 1999, The Flexibility of Fuzzy Clustering Illustrated by Examples , *Journal of Chemo metrics*, 13: 435-444.
- Ong, C.,S.; Lai, W.K., 2000, *Enhanced Password Authentication through Typing Biometrics with the K-Means Clustering Algorithm* , World Automation Congress, Seventh International Symposium on Manufacturing with Applications, 11-16 June, 2000, Maui, Hawaii
- Özdamar, K., 2002, *Paket Programlar ile İstatistiksel Veri Analizi-2*, Kaan Kitap evi, Eskişehir.
- Özekes, S., Veri Madenciliği Modelleri ve Uygulama Alanları, İstanbul Ticaret Odası Dergisi, <http://www.iticu.edu.tr/kutuphane/dergi/d3/M00041.pdf>
- Pal, N.R.; Biswas, J., 1997, Cluster Validation Using Grap Theoretic Concepts, *Pattern Recognition*, 30(6), 847-857.
- Pedrycz, W.; Gomide, F., 1998, *An Introduction To Fuzzy Sets: Analysis and Design*, Cambridge, MASS: MIT Press.
- Rezaee, R.; Lelieveldt, B.P.F.; Peiber, J.H.C., 1998, A New Cluster Validity Index for the Fuzzy c-Means, *Pattern Recognition Letters*, 19, 237-246.
- Saaty, T.L., 1980, *The Analytical Hierarchy Process*, McGraw Hill, New York.
- Şahin, M.; Hamarat, B., 2002, G-10 Avrupa Birliği ve OECD Ülkelerinin Sosyo-Ekonomik Benzerliklerinin Fuzzy Kümeleme Analizi ile Belirlenmesi, VI International Conference in Economics, 11-14, 2002, Ankara.
- Şahinli, F., Kümeleme Analizine Fuzzy Set Teorisi Yaklaşımı, Yüksek Lisans Tezi (İstatistik), Gazi Üniversitesi Fen bilimleri Enstitüsü , Ankara,1999.
- Şen, Z., 2001, *Bulanık Mantık ve Modelleme İlkeleri*, Bilge Sanat Yapım Evi, Ekim 2001, İstanbul.
- Sharma, S., 1996, *Applied Multivariate Techniques*, John Wiley & Sons, Inc. Newyork .
- Sheikholeslami, G.; Chatterjee, S.; Zhang, A., 1998, WaveCluster: A Multire Solution Clustering Approach for Very Large Spatial Databases, In *Proceedings of the 24th Conference on VLDB*, New York, 428-439.
- Smyth, P., 1996. Clustering using Monte Carlo Cross- Validation, In *Proceedings of KDD Conference*.
- Sugeno, M.; Yasukawa, T., 1993, A Fuzzy –Logic-Based Approach to Qualitative Modeling , *IEEE Trans. Fuzzy Systems*, vol. 1, no 1, pp.7-31.
- Takagi, T.; Sugeno, M., 1985, Fuzzy Identification of Systems and its Applications to Modeling and Control , *IEEE Trans. System .Man & Cybernet.* , vol. SMC-15 , pp.116-132.
- Tatlıdil, H., 2002, *Uygulamalı Çok Değişkenli İstatistiksel Analiz* , Akademi Matbaası, 4245, Ankara.
- Theodoridis, S., ; Koutroubas, K., 1999, *Pattern Recognition*, Academic Press.

Türkşen, B.; Bilgiç, T., Measurement of Membership Functions: Theoretical and Empirical Work, URL8.

Wang, W.; Muntz, R., 1997, STING : A Statistical Information Grid Approach to Spatial Data Mining. In Proceedings of 23rd VLDB Conference.

Xie, L.; Beni, G., 1991, A Validity Measure for Fuzzy Clustering, IEEE Transaction on Pattern Analysis and Machine Intelligence, 13(4), 841-846

Yılmaz, M.; Arslan, E., 2005, Bulanık Mantığın Jeodezik Problemlerin Çözümünde Kullanılması, 2. Mühendislik Ölçmeleri Sempozyumu, 23-25 Kasım, 2005, İTÜ, İstanbul.

Yu, D.; Chatterjee, S.; Sheikholeslami, G.; Zhang, A., 1998, Efficiently Detecting Arbitrary Shaped Clusters in Very Large Datasets with High Dimension, Department of Computer Science State University of New York at Buffalo, Buffalo, NY, 14260 USA

Zhang, H., 2005, A Note Fuzzy Clustering, Department of Computer Science and Engineering University Connecticut.

URL1: <http://www.mathworks.com/access/helpdesk/help/toolbox/fuzzy/>

URL2: <http://www.cems.uwe.ac.uk/~xzhang/PDF/MSc/Fuzzy%20Logic.pdf>.

URL3: <http://www.et.tudelft.nl/~babuska/transpfuzzmod.pdf>

URL4 : <http://www.yapay-zeka.org/modules/icontent/index.php?page=33>

URL5: <http://www.fmt.vein.hu/softcomp/fclusttoolbox/FuzzyClusteringToolbox.pdf>

URL6: <http://www.iticu.edu.tr/kutuphane/dergi/d3/M00041.pdf>

URL7: www.cs.sfu.ca/~han/bk/8clst.ppt

URL8: <http://www.ie.boun.edu.tr/~taner/publications/papers/membership.pdf>

EKLER

Ek Tablo 1. 1970-2002 Yıllarına İlişkin Türkiye Elektrik Tüketim Miktarları.

X	Y
7.307,80	8.289,30
8.289,30	9.527,30
9.527,30	10.530,10
10.530,10	11.358,70
11.358,70	13.491,70
13.491,70	16.068,90
16.068,90	17.968,80
17.968,80	18.933,80
18.933,80	19.663,10
19.663,10	20.398,20
20.398,20	22.030,00
22.030,00	23.586,80
23.586,80	24.465,10
24.465,10	27.635,20
27.635,20	29.708,60
29.708,60	32.209,70
32.209,70	36.697,30
36.697,30	39.721,50
39.721,50	43.120,00
43.120,00	48.633,60
48.633,60	50.295,70
50.295,70	54.613,10
54.613,10	60.406,30
60.406,30	61.420,30
61.420,30	67.092,30
67.092,30	74.326,80
74.326,80	81.884,90
81.884,90	87.704,60
87.704,60	91.201,90
91.201,90	98.295,70
98.295,70	97.070,00
97.070,00	102.800,00

Ek Tablo 2. Normalleştirilmiş Elektrik Tüketim Miktarları.

0,00	0,00
0,01	0,01
0,02	0,02
0,04	0,03
0,04	0,06
0,07	0,08
0,10	0,10
0,12	0,11
0,13	0,12
0,14	0,13
0,14	0,15
0,16	0,16
0,18	0,17
0,19	0,20
0,22	0,23
0,25	0,25
0,27	0,30
0,32	0,33
0,36	0,37
0,39	0,43
0,45	0,44
0,47	0,49
0,52	0,55
0,58	0,56
0,59	0,62
0,66	0,70
0,74	0,78
0,82	0,84
0,88	0,88
0,92	0,95
1,00	0,94
0,99	1,00

Ek 1: V_{sv} doğrulama kriterine göre en uygun küme sayısını bulan programın kaynak kodu:

```

unit opt;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, Grids, DB, DBTables, ExtCtrls, Math;
type
  TForm1 = class(TForm)
    Table1: TTable;
    Table1YT1: TFloatField;
    Table1YT2: TFloatField;
    Table1YT: TFloatField;
    Table1YILLAR: TFloatField;
    DataSource1: TDataSource;
    StringGrid1: TStringGrid;
    Label1: TLabel;
    Label2: TLabel;
    StringGrid2: TStringGrid;
    Label3: TLabel;
    StringGrid3: TStringGrid;
    Label4: TLabel;
    Edit1: TEdit;
    Label5: TLabel;
    Edit2: TEdit;
    Panel1: TPanel;
    Timer1: TTimer;
    Label6: TLabel;
    procedure FormCreate(Sender: TObject);
    procedure FormActivate(Sender: TObject);
    procedure Timer1Timer(Sender: TObject);
    procedure Panel1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  Form1: TForm1;
  i,g,q,b,k,j,l,s,r,copt,vs,tt,d,ss,ii,pp,t1,s1,k1,t2,s2,k2:integer;
  v4max,v4min,v0max,v0min,fa1,fa2:real;
  x1,x2,y,x11,x21,y1,x12,x22,y2,uz,mnn,max: Array[1..500] of real;
  u,u1:array[1..500,1..60,1..60] of real;
  veri:array[1..500,1..90] of real;
  veri1:array[1..500,1..90,1..60] of real;
  v,v1,ux,vx:array[1..500,1..90,1..60] of real;
  v2,rr:array[1..60,1..90] of real;
  v4,v0,v4n,v0n,svc:array[1..90] of real;

```

```

frk,fark1,fark2,mn,pcopt,degis,degis1,degis2:real;
degis3:array[1..500] of real;
uy:array[1..500,1..60] of real;
x:array[1..500,1..60] of real;
fark,frk1,vv:array[1..500,1..60] of real;
bolum:array[1..500,1..60] of real;
bolum1,amac2:real;
kare:array[1..500,1..60] of real;
carpim:array[1..60,1..60] of real;
toplam,min:array[1..60] of real;
amac,amac1:array[1..6000] of real;
amac3,vn:array[1..60] of real;
ll:array[1..500] of integer;
v4tp:array[1..500,1..100] of real;
dis:array[1..500] of real;
v4top:real;
mindis,dis1:real;
svcmin:real;
aa:array[1..500] of integer;
implementation
{$R *.dfm}
procedure TForm1.FormCreate(Sender: TObject);
begin
    Timer1.Enabled:=False;
    pp:=2;
    vs:=33;
    // ELEKTRİK TÜKETİM MİKTARLARININ GİRİLMESİ
    Table1.Open;
    for i:=1 to vs do
    begin
        x[i][1]:=StrToFloat(Table1YT1.Text);
        x[i][2]:=StrToFloat(Table1YT.Text);
        Table1.Next;
    end;
    Table1.Close;
    for i:=1 to vs do
    begin
        StringGrid1.Cells[1,i]:=FloatToStr(x[i][1]);
        StringGrid1.Cells[2,i]:=FloatToStr(x[i][2]);
    end;
    // MAKSİMUM VE MİNİMUM DEĞERLERİN BULUNMASI
    for i:=1 to pp do
        max[i]:=-1000000000;
    for i:=1 to vs do
    for j:=1 to pp do
    if max[j]<x[i][j] then max[j]:=x[i][j];

    for i:=1 to pp+1 do

```



```

mnn[i]:=1000000000000;
for i:=1 to vs do
for j:=1 to pp do
if x[i][j]<mnn[j] then mnn[j]:=x[i][j];
// VERİLERİN NORMALLEŞTİRİLMESİ
for i:=1 to vs do
begin
for j:=1 to pp do
veri[i][j]:=(x[i][j]-mnn[j])/(max[j]-mnn[j]);
end;
b:=1; q:=2; copt:=2; g:=0; frk:=0;
vs:=21;
end;
procedure TForm1.FormActivate(Sender: TObject);
begin
Form1.WindowState:=wsNormal;
//BAŞLANGIÇ DEĞERLERİNİN GİRİLMESİ
q:=q+1; frk:=0; fark1:=0;
for i:=1 to 100 do
toplam[i]:=0;
for i:=1 to 1000 do
amac[i]:=0;
bolum1:=0;
Timer1.Enabled:=True;
StringGrid2.ColCount:=q+1;
StringGrid2.DefaultColWidth:=90;
StringGrid2.Width:=(q+1)*90;
StringGrid3.RowCount:=q+1;
StringGrid3.Height:=(q+1)*35;

randomize;
for i:=1 to 100 do
for j:=1 to pp do
v[i][j][q]:=random;
for i:=1 to q do
StringGrid2.Cells[i,0]:='U'+IntToStr(i);
For i:=1 to vs do
StringGrid2.Cells[0,i]:=IntToStr(i);
for i:=1 to q do
StringGrid3.Cells[0,i]:='V'+IntToStr(i);
for i:=1 to pp do
StringGrid3.Cells[i,0]:='X'+IntToStr(i);
for i:=1 to q do
for j:=1 to pp do
StringGrid3.Cells[j,i]:= FloatToStr(v[i][j][q]);
g:=0;
amac[g+1]:=0;

```

```

        frk:=0;
        fark1:=0;
        for i:=1 to q do
            for j:=1 to pp do   carpim[i][j]:=0;
            for i:=1 to q do   toplam[i]:=0;
            Form1.WindowState:=wsMaximized;
            Timer1.Enabled:=True;
        end;
    procedure TForm1.Timer1Timer(Sender: TObject);
    begin
        g:=g+1;
        //Üyelik değerlerinin Bulunması
        for i:=1 to vs do
            for s:=1 to q do
                begin
                    for j:=1 to q do
                        begin
                            for k:=1 to pp do
                                frk:=frk+sqr(veri[i][k]-v[j][k][q]);
                                fark[i][j]:=sqrt(frk);
                                frk:=0;
                                bolum[i][j]:=Power(fark[i][s]/fark[i][j],2);
                                bolum1:=bolum1+bolum[i][j];
                            end;
                            u[i][s][q]:=1/bolum1;
                            StringGrid2.Cells[s,i]:=FloatToStrF(u[i][s][q],ffgeneral,1,5);
                            bolum1:=0;
                        end;
                    for i:=1 to q do
                        for j:=1 to pp do
                            v1[i][j][q]:=v[i][j][q];
                    for k:=1 to q do
                        for j:=1 to pp do
                            begin
                                carpim[k][j]:=0;
                            end;
                    for i:=1 to q do toplam[i]:=0;
                    for i:=1 to vs do
                        for j:=1 to q do
                            kare[i][j]:=Power(u[i][j][q],2);
                    for k:=1 to q do
                        for j:=1 to pp do
                            for i:=1 to vs do   carpim[k][j]:=carpim[k][j]+kare[i][k]*veri[i][j];
                    for k:=1 to q do
                        for i:=1 to vs do   toplam[k]:=toplam[k]+kare[i][k];

        // Küme Merkezlerinin Bulunması

```

```

for i:=1 to q do
for k:=1 to pp do
begin
  v[i][k][q]:=carpim[i][k]/toplam[i];
  StringGrid3.Cells[k,i]:=FloatToStrF(v[i][k][q],ffgeneral,1,5);
end;
for i:=1 to q do
for j:=1 to pp do carpim[i][j]:=0;
for i:=1 to q do toplam[i]:=0;
dis1:=0;
for i:=1 to q do
for j:=1 to pp do
  dis1:=dis1+sqr(v[i][j][q]-v1[i][j][q]);
  dis1:=sqrt(dis1);
if dis1<=0.0000000000000001 then
begin
  Timer1.Enabled:=False;
  amac3[q]:=amac[g];
  //*****
  for i:=1 to vs do
  for j:=1 to q do
  ux[i][j][q]:=u[i][j][q];
  for i:=1 to q do
  for j:=1 to pp do
  vx[i][j][q]:=v[i][j][q];

for i:=1 to q do aa[i]:=0;
  for j:=1 to q do
  begin
    for i:=1 to vs do
    if ll[i]=j then
    begin
      aa[j]:=aa[j]+1;
      for k:=1 to pp do
      begin
        veri1[aa[j]][k][j]:=veri[i][k];
      end;
    end;
  end;
end;
//-----
// Vu ve Vo deęerlerinin bulunması
v4top:=0;v4[q]:=0;
for i:=1 to q do vn[i]:=0;
for i:=1 to vs do
for j:=1 to q do vv[i][j]:=0;
for j:=1 to q do
begin
for i:=1 to aa[j] do

```

```

begin
  for k:=1 to pp do
    begin
      vv[i][j]:=vv[i][j]+sqr(veri1[i][k][j]-v[j][k][q]);
    end;
    vv[i][j]:=sqr(vv[i][j]);
    vn[j]:=vn[j]+vv[i][j]
  end;
  vn[j]:=vn[j]/aa[j];
  v4top:=v4top+vn[j];
end;
v4[q]:=(v4top/q);
Edit1.Text:=FloatToStr(v4[q]);
//-----
//-----
v0[q]:=0;
k:=0;
for i:=1 to 100 do dis[i]:=0;
for i:=1 to q-1 do
  for j:=i+1 to q do
    begin
      k:=k+1;
      for tt:=1 to pp do
        dis[k]:=dis[k]+sqr(v[i][tt][q]-v[j][tt][q]);
        dis[k]:=sqr(dis[k]);
      end;
    mindis:=4000000000000000;
    for i:=1 to k do
      if dis[i]<mindis then mindis:=dis[i];
    v0[q]:=q/mindis;
    k:=0;
    Edit2.Text:=FloatToStr(v0[q]);
    if q=9 then
      begin
        //-----
        v4max:=-100000000000; v4min:=100000000000000;
        for i:=2 to 9 do
          if v4max<v4[i] then v4max:=v4[i];
          for i:=2 to 9 do
            if v4[i]<v4min then v4min:=v4[i];
          v0max:=-1000000000; v0min:=100000000000;
          for i:=2 to 9 do
            if v0max<v0[i] then v0max:=v0[i];
            for i:=2 to 9 do
              if v0[i]<v0min then v0min:=v0[i];
            for i:=2 to 9 do
              begin
                v4n[i]:=0;v0n[i]:=0;svc[i]:=0;

```

```

end;
for i:=2 to 9 do
begin
    fa1:=(v4[i]-v4min);
    fa2:=(v4max-v4min);
    v4n[i]:=(fa1/fa2);
    fa1:=0;
    fa2:=0;
    fa1:=(v0[i]-v0min);
    fa2:=(v0max-v0min);
    v0n[i]:=fa1/fa2;
    fa1:=0;
    fa2:=0;
end;
for i:=2 to 9 do svc[i]:=0;
for i:=2 to 9 do
svc[i]:=v4n[i]+v0n[i];
//Vsv DEĞERİNİN BULUNMASI
svcmin:=100000000;
for i:=2 to 9 do
if svc[i]<svcmin then
begin
    svcmin:=svc[i];
    copt:=i;
    r:=copt;
    amac2:=amac3[i];
end;
StringGrid2.ColCount:=copt+1;
StringGrid2.DefaultColWidth:=70;
StringGrid2.Width:=(copt+1)*70;
StringGrid3.RowCount:=copt+1;
StringGrid3.Height:=(copt+1)*35;
for i:=1 to copt+1 do
    StringGrid2.Cells[i,0]:='U'+IntToStr(i);
for i:=1 to copt+1 do
    StringGrid3.Cells[0,i]:='V'+IntToStr(i);
for i:=1 to r do
for j:=1 to pp do
StringGrid3.Cells[j,i]:=FloatToStrF(vx[i][j][r],ffgeneral,1,5);
for i:=1 to vs do
for j:=1 to r do
StringGrid2.Cells[j,i]:=FloatToStrF(ux[i][j][r],ffgeneral,1,5);
Label6.Caption:=IntToStr(copt)+' optimal cluster number';
end else
begin
    dis1:=0; g:=0;
    Form1.Activate;
end;

```

```
end;  
end;  
procedure TForm1.Panel1Click(Sender: TObject);  
begin  
halt;  
end;  
end.
```

Ek 2: BCO Kümeleme algoritmasının program kaynak kodu:

```

unit bco;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Grids, StdCtrls, ExtCtrls, DB, DBTables;
type
  TForm1 = class(TForm)
    Label1: TLabel;
    StringGrid1: TStringGrid;
    Label2: TLabel;
    StringGrid2: TStringGrid;
    Label3: TLabel;
    StringGrid3: TStringGrid;
    Label4: TLabel;
    Label5: TLabel;
    Edit1: TEdit;
    Label6: TLabel;
    Edit2: TEdit;
    Button1: TButton;
    Table1: TTable;
    Table1YT1: TFloatField;
    Table1YT2: TFloatField;
    Table1YT: TFloatField;
    Table1YILLAR: TFloatField;
    DataSource1: TDataSource;
    Table2: TTable;
    Table2U1: TFloatField;
    Table2U2: TFloatField;
    Table2U3: TFloatField;
    Table2U4: TFloatField;
    DataSource2: TDataSource;
    Timer1: TTimer;
  procedure FormCreate(Sender: TObject);
  procedure FormActivate(Sender: TObject);
  procedure Timer1Timer(Sender: TObject);
  procedure Button1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;
  i,j,vs,p,q,s,k,tt,ss,t,f,a:integer;

```

```

x,veri,u,v,v3,bolum,carpim,fark,v1,kare,vv,toplamv,g,b,tt,z,zz,zzz,ters,wt,w,wwt,
di,d1,veri1,uk,ug,u1:array[1..300,1..300] of double;
max,toplam,mn,mxq,v4tp,v4,vn,v0,dis1,std,toplamm,tu,st,yt,mutlakfark,gg,uyy:array
[1..300] of real;
uy2,uu:array[1..100,1..100,1..100] of real;
frk,fark1,bolum1,dis,mindis,ortfark,frk1,us,m,fark2,amac:real;

```

implementation

```
{ $R *.dfm }
```

```
procedure TForm1.FormCreate(Sender: TObject);
```

```
begin
```

```
  Form1.WindowState:=wsMaximized;
```

```
  randomize;
```

```
  vs:=33; p:=2;q:=4;ss:=0;
```

```
  Timer1.Enabled:=False;
```

```
  // ELEKTTRİK TÜKETİM DEĞERLERİNİN GİRİLMESİ
```

```
Table1.Open;
```

```
for i:=1 to vs do
```

```
begin
```

```
  x[i][1]:=StrToFloat(Table1.YT1.Text);
```

```
  x[i][2]:=StrToFloat(Table1.YT2.Text);
```

```
  Table1.Next;
```

```
end;
```

```
Table1.Close;
```

```
for i:=1 to vs do
```

```
begin
```

```
  StringGrid3.Cells[1,i]:=FloatToStr(x[i][1]);
```

```
  StringGrid3.Cells[2,i]:=FloatToStr(x[i][2]);
```

```
end;
```

```
for i:=1 to p do mn[i]:=100000000000;
```

```
for i:=1 to vs do
```

```
for j:=1 to p do
```

```
if x[i][j]<mn[j] then mn[j]:=x[i][j];
```

```
max[1]:=190000;max[2]:=190000;
```

```
for i:=1 to vs do
```

```
begin
```

```
  for j:=1 to p do
```

```
    veri[i][j]:=(x[i][j]-mn[j])/(max[j]-mn[j]);
```

```
end;
```

```
for i:=1 to q do
```

```
for j:=1 to p do
```

```
v[i][j]:=random;
```

```
frk:=0;
```

```
fark1:=0;
```



```

    for i:=1 to vs do toplam[i]:=0;
    for i:=1 to vs do
    for j:=1 to p do carpim[i][j]:=0;
    bolum1:=0; a:=0;
    vs:=21;m:=2;
end;

procedure TForm1.FormActivate(Sender: TObject);
begin
    Timer1.Enabled:=True;
end;

procedure TForm1.Timer1Timer(Sender: TObject);
begin
    frk:=0;
    a:=a+1;
    Edit1.Text:=IntToStr(a) ;
    for i:=1 to vs do
    begin
    for s:=1 to q do
    begin
    for j:=1 to q do
    begin
        for k:=1 to p do frk:=frk+sqr(veri[i][k]-v[j][k]);
        fark[i][j]:=sqr(frak);
        frk:=0;
        bolum[i][j]:=power(fark[i][s]/fark[i][j],2/(m-1));
        bolum1:=bolum1+(bolum[i][j]);
    end;
    u[i][s]:=1/bolum1;
    StringGrid1.Cells[s,i]:=FloatToStrF(u[i][s],ffgeneral,1,6);
    bolum1:=0;
    end;
end;

fark1:=0;
fark2:=0;
amac:=0;
for j:=1 to q do
begin
    for i:=1 to vs do
    begin
        for k:=1 to p do
        begin
            fark1:=fark1+sqr(veri[i][k]-v[j][k]);
        end;
        fark2:=fark1;
        amac:=amac+power(u[i][j],m)*fark2;
    end;
end;

```

```

        fark1:=0;
        fark2:=0;
    end;
end;
    Edit2.Text:=FloatToStr(amac) ;

for i:=1 to q do
for j:=1 to p do v1[i][j]:=0;

for i:=1 to q do
for j:=1 to p do v1[i][j]:=v[i][j];

for k:=1 to q do
for j:=1 to p do
begin
    carpim[k][j]:=0;
end;
for i:=1 to q do toplam[i]:=0;

for i:=1 to vs do
for j:=1 to q do
    kare[i][j]:=0;
for i:=1 to vs do
for j:=1 to q do
    kare[i][j]:=power(u[i][j],m);

for k:=1 to q do
for i:=1 to vs do
for j:=1 to p do
    carpim[k][j]:=carpim[k][j]+kare[i][k]*veri[i][j];

for k:=1 to q do
for i:=1 to vs do toplam[k]:=toplam[k]+kare[i][k];
for i:=1 to q do
for j:=1 to p do
    v[i][j]:=0;
for i:=1 to q do
begin
    for j:=1 to p do
    begin
        v[i][j]:=carpim[i][j]/toplam[i];
        StringGrid2.Cells[j,i]:=FloatToStrF(v[i][j],ffgeneral,1,5);
    end;
end;
for i:=1 to q do
for j:=1 to p do carpim[i][j]:=0;
for i:=1 to q do toplam[i]:=0;
dis:=0;

```

```
for i:=1 to q do
for j:=1 to p do
    dis:=dis+sqr(v[i][j]-v1[i][j]);
    dis:=sqr(dis);
if dis<0.0000000000001 then
begin
    Timer1.Enabled:=False;
    showmessage('İşleminiz Bitmiştir.') ;
end;
end;
procedure TForm1.Button1Click(Sender: TObject);
begin
    halt;
end;
end.
```

Ek 3: BCRM Algoritmasının program kaynak kodu:

```
unit Unit1;

interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, DB, DBTables, ExtCtrls, Grids, StdCtrls, math,
  OleCtrls, Wintypes, Winprocs;

type
  TForm1 = class(TForm)
    Panel1: TPanel;
    Edit1: TEdit;
    Label1: TLabel;
    Label2: TLabel;
    Edit2: TEdit;
    Label3: TLabel;
    Edit3: TEdit;
    Panel2: TPanel;
    StringGrid1: TStringGrid;
    StringGrid2: TStringGrid;
    Label4: TLabel;
    Label5: TLabel;
    Timer1: TTimer;
    Table1: TTable;
    DataSource1: TDataSource;
    Edit4: TEdit;
    Edit5: TEdit;
    StringGrid3: TStringGrid;
    Table1YT1: TFloatField;
    Table1YT: TFloatField;
    Label6: TLabel;
    Label7: TLabel;
    Table1YT2: TFloatField;
    Timer2: TTimer;
    StringGrid4: TStringGrid;
    Label8: TLabel;
    Table2: TTable;
    Table2Y: TFloatField;
    DataSource2: TDataSource;
    Button1: TButton;
    Table3: TTable;
    Table3Y1: TFloatField;
    Table3Y2: TFloatField;
    Table3Y3: TFloatField;
    Table3Y4: TFloatField;
    DataSource3: TDataSource;
```

```

Table3G: TFloatField;
Table4: TTable;
Table4U1: TFloatField;
Table4U2: TFloatField;
Table4U3: TFloatField;
Table4U4: TFloatField;
DataSource4: TDataSource;
procedure FormCreate(Sender: TObject);
procedure Timer1Timer(Sender: TObject);
procedure FormActivate(Sender: TObject);
procedure Panel2Click(Sender: TObject);
procedure Timer2Timer(Sender: TObject);
procedure Button1Click(Sender: TObject);
private
  procedure Tersmatrix;
  { Private declarations }
public

  { Public declarations }
end;
var
  Form1: TForm1;
  i,j,k,t,s,n,d,q,c,p,vs,g,h,r,tt:integer;
x,veri,y,z,yy,o,bolum,fark,v,x1,uy,v2,toplamv,stt,zz,zzz,ty,gy,tyi,us,vu2,z1,dat:array[
1..300,1..300] of real;
max,mn,det,yt,ut,topyy,toplamm,std,st,mxq,vu1,dis2,oyi,ust,yt1,tpu,vu3,mutlakfark,
mx,dist,y1,y2,y3,y4:array[1..1500] of real;
u:array[1..300,1..10,1..1500] of real;
b,a,det1,ters:array[1..10,1..10,1..10]of double;
amac,mean,dis1,bolum1,vu,vo,mindis,vu4,ortfark,m,m1,mx1:real;
ke,kgi,ll,aa:array[1..300] of integer;
implementation {$R *.dfm}
uses matrix ;
const
  MatSize = 2;
var
  Mat1: TMatrix;

procedure TForm1.tersmatrix;
begin
  Mat1 := TMatrix.Create (nil);
  Mat1.Resize(matsize,matsize);
  for i:=1 to matsize do
  for j:=1 to matsize do
  Mat1.Elem[i,j]:=a[i][j][c];
  if Mat1.Invert then
  for i:=1 to matsize do
  for j:=1 to matsize do

```

```

ters[i][j][c]:=Mat1.Elem[i,j];
end;
procedure TForm1.FormCreate(Sender: TObject);
begin
Form1.WindowState:=wsMaximized;
Timer1.Enabled:=False;
Timer2.Enabled:=False;

vs:=32; q:=4; p:=1;
r:=32;
Table1.open;
for i:=1 to vs do
begin
x[i][1]:=StrToFloat(Table1.YT1.Text);
x[i][2]:=StrToFloat(Table1.YT2.Text);
Table1.Next;

end;
Table1.Close;
//randomize;
for i:=1 to vs do
for j:=1 to q do
u[i][j][1]:=random;
for i:=1 to vs do
for j:=1 to q do
StringGrid1.Cells[j,i]:=floatToStr(u[i][j][1]);
for i:=1 to 100 do
max[i]:=-10000;
for i:=1 to vs do
for j:=1 to p+1 do
if max[j]<x[i][j] then max[j]:=x[i][j];
for i:=1 to 100 do
mn[i]:=1000000;

for i:=1 to vs do
for j:=1 to p+1 do
if x[i][j]<mn[j] then mn[j]:=x[i][j];
max[1]:=190000;max[2]:=190000;
for i:=1 to vs do
begin
for j:=1 to p+1 do
veri[i][j]:=(x[i][j]-mn[j])/(max[j]-mn[j]);
end;
amac:=0;
StringGrid1.Cells[1,0]:='U1';
StringGrid1.Cells[2,0]:='U2';
StringGrid1.Cells[3,0]:='U3';

StringGrid2.Cells[1,0]:='A0';

```

```

StringGrid2.Cells[2,0]:='A1';
StringGrid2.RowCount:=q+1;
for i:=1 to vs do
StringGrid1.Cells[0,i]:=IntToStr(i);
Timer1.Enabled:=False;
for i:=1 to vs do
x1[i][1]:=1;

for j:=1 to vs do
for i:=1 to p do
x1[j][i+1]:=veri[j][i];
g:=1;
vs:=20;
r:=32;
m:=2.0;
end;

procedure TForm1.Timer1Timer(Sender: TObject);
begin
Edit1.Text:=IntToStr(g);
//1.SATIR
for i:=1 to q do
for j:=1 to p+1 do
a[1][j][i]:=0;

for j:=1 to p+1 do
for k:=1 to vs do
for i:=1 to q do
a[1][j][i]:=a[1][j][i]+power(u[k][i][g],m)*x1[k][j];

//2.SATIR
for i:=1 to q do
for j:=1 to p+1 do
a[2][j][i]:=0;
for j:=1 to p+1 do
for k:=1 to vs do
for i:=1 to q do
a[2][j][i]:=a[2][j][i]+power(u[k][i][g],m)*x1[k][j]*veri[k][1];

//Y MATRİSİNİN OLUŞTURULMASI
for i:=1 to q do
for j:=1 to p+1 do
y[i][j]:=0;

for k:=1 to vs do
for i:=1 to q do
for j:=1 to p+1 do
y[i][j]:=y[i][j]+power(u[k][i][g],m)*veri[k][p+1]*x1[k][j];

```

```

for i:=1 to p+1 do
for j:=1 to q do z[j][i]:=0;

for c:=1 to q do
for i:=1 to p+1 do
for k:=1 to p+1 do
    z[c][i]:=z[c][i]+(y[c][k]*ters[k][i][c]);

for i:=1 to q do
for j:=1 to p+1 do
    StringGrid2.Cells[j,i]:=FloatToStrF(z[i][j],ffgeneral,10,5);

//Clusterlere Göre Y değerlerinin Bulunması
for i:=1 to vs do
for j:=1 to q do
    o[i][j]:=0;
for i:=1 to vs do
for j:=1 to q do
for k:=1 to p+1 do
o[i][j]:=o[i][j]+z[j][k]*x1[i][k];

//Üyelik Fonksiyon değerlerinin bulunması
m1:=0;
m1:=2/(m-1);
bolum1:=0;
g:=g+1;
bolum1:=0;
for i:=1 to vs do
begin
    for s:=1 to q do
    begin
        for j:=1 to q do
        begin
            fark[i][j]:=sqr(veri[i][p+1]-o[i][j]);
            bolum[i][j]:=sqrt(fark[i][s]/fark[i][j]);
            bolum1:=bolum1+power(bolum[i][j],m1);
        end;
        u[i][s][g]:=1/bolum1;
        StringGrid1.Cells[s,i]:=FloatToStrF(u[i][s][g],ffgeneral,1,5);
        bolum1:=0;
    end;
end;
dis1:=0;
for i:=1 to vs do
for j:=1 to q do

```



```

begin
  dis1:=dis1+sqr(u[i][j][g]-u[i][j][g-1]);
end;
dis1:=sqrt(dis1);
Edit2.Text:=FloatToStrF(dis1,ffgeneral,1,5);
if dis1<0.0000000001 then
begin
  Timer1.Enabled:=False;
  Showmessage('İşleminiz Bitmiştir');

vs:=32;
for i:=1 to vs do
for j:=1 to q do
  o[i][j]:=0;
for i:=1 to vs do
for j:=1 to q do
for k:=1 to p+1 do
o[i][j]:=o[i][j]+z[j][k]*x1[i][k];

//Üyelik Fonksiyon değerlerinin bulunması

m1:=0;
m1:=2/(m-1);
bolum1:=0;
g:=g+1;
bolum1:=0;
for i:=1 to vs do
begin
  for s:=1 to q do
  begin
    for j:=1 to q do
    begin
      fark[i][j]:=sqr(veri[i][p+1]-o[i][j]);
      bolum[i][j]:=sqrt(fark[i][s]/fark[i][j]);
      bolum1:=bolum1+power(bolum[i][j],m1);
    end;
    u[i][s][g]:=1/bolum1;
    StringGrid1.Cells[s,i]:=FloatToStr(u[i][s][g]);
    bolum1:=0;
  end;
end;
end;
Table4.Open;
for i:=1 to 20 do
begin
  Table4.Append;
  Table4U1.Text:=FloatToStr(u[i][1][g]);
  Table4U2.Text:=FloatToStr(u[i][2][g]);
  Table4U3.Text:=FloatToStr(u[i][3][g]);

```

```

Table4U4.Text:=FloatToStr(u[i][4][g]);
Table4.Next;

end;
Table4.Close;

for i:=1 to vs do ut[i]:=0;
for i:=1 to vs do
for j:=1 to q do
    ut[i]:=ut[i]+u[i][j][g];

for i:=1 to vs do
for j:=1 to q do yy[i][j]:=0;

for i:=1 to vs do
for j:=1 to q do
for k:=1 to p+1 do
yy[i][j]:=yy[i][j]+z[j][k]*x1[i][k];
for i:=1 to vs do mx[i]:=0;

for i:=1 to q do
aa[i]:=0;

for i:=1 to vs do
for j:=1 to q do
begin
    if mx[i]<u[i][j][g] then
    begin
        mx[i]:=u[i][j][g];
        ll[i]:=j;
    end;
end;
for j:=1 to q do
for i:=1 to vs do
begin
    if ll[i]=j then
    begin
        aa[j]:=aa[j]+1;
        dat[aa[j]][j]:=veri[i][2];
    end;
end;
for i:=1 to vs do
for j:=1 to q do
stringGrid1.Cells[j,i]:='          ';
for j:=1 to q do
for i:=1 to aa[j] do
stringGrid1.Cells[j,i]:=FloatToStr(dat[i][j]);
for i:=1 to vs do

```

```

topyy[i]:=0;
vs:=32;
for i:=1 to vs do
topyy[i]:=yy[i][ll[i]];

for i:=1 to vs do yt[i]:=0;

for i:=1 to vs do
yt[i]:=topyy[i];

//Eğitim Verileri için hatanın bulunması
for i:=1 to vs do
mutlakfark[i]:=abs(x[i][p+1]-(yt[i]*(max[2]-mn[2])+mn[2]))/x[i][p+1]);
vs:=20;
ortfark:=0;
for i:=1 to vs do
ortfark:=ortfark+mutlakfark[i]*100;
ortfark:=ortfark/vs;
Edit3.Text:=FloatToStr(ortfark);
//Doğrulama Verileri için Hatanın Bulunması
ortfark:=0;
for i:=21 to 28 do
ortfark:=ortfark+mutlakfark[i]*100;
ortfark:=ortfark/8;
Edit4.Text:=FloatToStr(ortfark);
//Test Verileri için Hatanın Bulunması
ortfark:=0;
for i:=29 to 32 do
ortfark:=ortfark+mutlakfark[i]*100;
ortfark:=ortfark/4;
Edit5.Text:=FloatToStr(ortfark);
vs:=32;
for i:=1 to vs do
begin
StringGrid4.Cells[0,i]:=IntToStr(1970+i);
StringGrid4.Cells[1,i]:=FloatToStr(mutlakfark[i]*100);
end;
vs:=20;

for i:=1 to vs do
for j:=1 to q do
StringGrid1.Cells[j,i]:= ' ';
StringGrid1.ColCount:=3;
StringGrid1.Cells[1,0]:='GERÇEKY';
StringGrid1.Cells[2,0]:='TAHMİNY';
Timer2.Enabled:=true;
vs:=32;
for i:=1 to vs do

```

```

begin
  StringGrid1.Cells[1,i]:=FloatToStr(x[i][p+1]);
  StringGrid1.Cells[2,i]:=FloatToStr((yt[i]*(max[2]-mn[2])+mn[2]));

  end;
end;
dis1:=0;

end;
procedure TForm1.FormActivate(Sender: TObject);
begin
  Timer1.Enabled:=True;
end;
procedure TForm1.Panel2Click(Sender: TObject);
begin
  halt;
end;

procedure TForm1.Timer2Timer(Sender: TObject);
begin
  //TAHMİN DEĞERLERİNİN HESAPLANMASI
  r:=r+1;
  veri[r][1]:=veri[r-1][2];
  x1[r][1]:=1;
  x1[r][2]:=veri[r][1];
  for j:=1 to q do o[r][j]:=0;
  for j:=1 to q do
    for k:=1 to p+1 do o[r][j]:=o[r][j]+z[j][k]*x1[r][k];

  for j:=1 to q do dist[j]:=0;

  for j:=1 to q do
    for i:=1 to aa[j] do
      begin
        dist[j]:=dist[j]+sqr(o[r][j]-dat[i][j]);
        dist[j]:=sqrt(dist[j]);
      end;
  mx1:=1000000000000;

  for j:=1 to q do
    if mx1>dist[j] then
      begin
        mx1:=dist[j];
        veri[r][2]:=o[r][j];
      end;
  end;

  if r=40 then
    begin

```

```

Timer2.Enabled:=False;
for i:=33 to r do
yt[i]:=veri[i][2];
for i:=33 to r do
StringGrid4.Cells[1,i-32]:=FloatToStr(veri[i][2]*(max[2]-mn[2])+mn[2]);
end;
end;
procedure TForm1.Button1Click(Sender: TObject);
begin
vs:=40;
for i:=1 to vs do
y1[i]:=z[1][1]+z[1][2]*veri[i][1];

for i:=1 to vs do
y2[i]:=z[2][1]+z[2][2]*veri[i][1];
for i:=1 to vs do
y3[i]:=z[3][1]+z[3][2]*veri[i][1];
for i:=1 to vs do
y4[i]:=z[4][1]+z[4][2]*veri[i][1];
for i:=1 to vs do
begin
y1[i]:=y1[i]*(max[2]-mn[2])+ mn[2];
y2[i]:=y2[i]*(max[2]-mn[2])+ mn[2];
y3[i]:=y3[i]*(max[2]-mn[2])+ mn[2];
y4[i]:=y4[i]*(max[2]-mn[2])+ mn[2];

end;
end;
end.

```

Ek 4: Takagi-Sugeno Modelleme Yöntemi Kaynak Kodu:

```

unit takagi-sugeno;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Grids, StdCtrls, DB, DBTables, ExtCtrls, math;
type
  TForm1 = class(TForm)
    Label1: TLabel;
    StringGrid1: TStringGrid;
    Label2: TLabel;
    StringGrid2: TStringGrid;
    Timer1: TTimer;
    Table1: TTable;
    Table1YT1: TFloatField;
    Table1YT: TFloatField;
    DataSource1: TDataSource;
    Panel1: TPanel;
    Panel2: TPanel;
    Label3: TLabel;
    Edit1: TEdit;
    Label4: TLabel;
    StringGrid3: TStringGrid;
    Panel3: TPanel;
    Label5: TLabel;
    Edit2: TEdit;
    Label6: TLabel;
    Edit3: TEdit;
    Table1YT2: TFloatField;
    Table2: TTable;
    Table2U1: TFloatField;
    Table2U2: TFloatField;
    DataSource2: TDataSource;
    Table3: TTable;
    Table3V1: TFloatField;
    DataSource3: TDataSource;
    Table4: TTable;
    Table4S1: TFloatField;
    Table4S2: TFloatField;
    DataSource4: TDataSource;
    Table3V2: TFloatField;
    Timer2: TTimer;
  procedure FormCreate(Sender: TObject);
  procedure FormActivate(Sender: TObject);
  procedure Timer1Timer(Sender: TObject);
  procedure Panel2Click(Sender: TObject);
  procedure Panel1Click(Sender: TObject);
  end;

```

```

    procedure Panel3Click(Sender: TObject);
    procedure Timer2Timer(Sender: TObject);
private
    procedure Tersmatrix;
    { Private declarations }
public
    { Public declarations }
end;
var
    Form1: TForm1;
    i,j,vs,p,q,s,k,tt,ss,t,f,a:integer;
x,veri,u,v,v3,bolum,carpim,fark,v1,kare,vv,toplamv,g,b,stt,z,zz,zzz,ters,wt,w,wwt,di,
d1,veri1,uk,ug,u1:array[1..300,1..300] of double;
max,toplam,mn,mxq,v4tp,v4,vn,v0,dis1,std,toplamm,tu,st,yt,mutlakfark,gg,uyy,max
u:array[1..300] of real;
    uy2,uu,xx:array[1..100,1..100,1..100] of real;
    frk,fark1,bolum1,dis,mindis,ortfark,frk1,us,m:real;
    l:array[1..100] of integer;
implementation uses ts2, matrix ;
    const
        MatSize =4;
var
    Mat1: TMatrix;
procedure TForm1.tersmatrix;
begin
    Mat1 := TMatrix.Create (nil);
    Mat1.Resize(matsize,matsize);
    for i:=1 to matsize do
    for j:=1 to matsize do
    Mat1.Elem[i,j]:=wwt[i][j];
    for i:=1 to matsize do
    for j:=1 to matsize do
    ters[i][j]:=0;
    if Mat1.Invert then
    for i:=1 to matsize do
    for j:=1 to matsize do
    ters[i][j]:=Mat1.Elem[i,j];
    end;
    {$R *.dfm}
procedure TForm1.FormCreate(Sender: TObject);
begin
    Form1.WindowState:=wsMaximized;
    a:=0;
    randomize;
    vs:=32; p:=2;q:=2;
    ss:=0;
    Timer1.Enabled:=False;

```

```

Table1.Open;
for i:=1 to vs do
begin
  x[i][1]:=StrToFloat(Table1.YT1.Text);
  x[i][2]:=StrToFloat(Table1.YT2.Text);
  Table1.Next;
end;
Table1.Close;
for i:=1 to p do mn[i]:=100000000000;
for i:=1 to vs do
for j:=1 to p do
if x[i][j]<mn[j] then
mn[j]:=x[i][j];
max[1]:=1900000;max[2]:=190000;
for i:=1 to vs do
begin
  for j:=1 to p do
    veri[i][j]:=(x[i][j]-mn[j])/(max[j]-mn[j]);
end;
for i:=1 to q do
for j:=1 to p do
v[i][j]:=random;
frk:=0;
fark1:=0;
for i:=1 to vs do
toplam[i]:=0;
for i:=1 to vs do
for j:=1 to p do carpim[i][j]:=0;
bolum1:=0;
vs:=20;
m:=1.5;
Timer2.Enabled:=False;
end;
procedure TForm1.FormActivate(Sender: TObject);
begin
Timer1.Enabled:=True;
end;
procedure TForm1.Timer1Timer(Sender: TObject);
begin
frk:=0;
for i:=1 to vs do
begin
for s:=1 to q do
begin
for j:=1 to q do
begin
for k:=1 to p do

```



```

    frk:=frk+sqr(veri[i][k]-v[j][k]);
    fark[i][j]:=sqr(frk);
    frk:=0;
    bolum[i][j]:=power(fark[i][s]/fark[i][j],2/(m-1));
    bolum1:=bolum1+(bolum[i][j]);
  end;
  u[i][s]:=1/bolum1;
  StringGrid1.Cells[3,i-1]:=IntToStr(i);
  StringGrid1.Cells[s-1,i-1]:=FloatToStrF(u[i][s],ffgeneral,1,6);
  bolum1:=0;
end;
end;
for i:=1 to q do
for j:=1 to p do
  v1[i][j]:=0;
for i:=1 to q do
for j:=1 to p do
  v1[i][j]:=v[i][j];
  for k:=1 to q do
  for j:=1 to p do
  begin
    carpim[k][j]:=0;
  end;
  for i:=1 to q do toplam[i]:=0;
  for i:=1 to vs do
  for j:=1 to q do
    kare[i][j]:=0;
  for i:=1 to vs do
  for j:=1 to q do
    kare[i][j]:=power(u[i][j],m);
  for k:=1 to q do
  for i:=1 to vs do
  for j:=1 to p do
  carpim[k][j]:=carpim[k][j]+kare[i][k]*veri[i][j];
  for i:=1 to vs do
  for j:=1 to p do
  StringGrid1.Cells[j-1,i-1]:=FloatToStr(veri[i][j]) ;
  for k:=1 to q do
  for i:=1 to vs do toplam[k]:=toplam[k]+kare[i][k];
  for i:=1 to q do
  for j:=1 to p do
    v[i][j]:=0;
  for i:=1 to q do
  begin
  for j:=1 to p do
  begin
    v[i][j]:=carpim[i][j]/toplam[i];
    StringGrid2.Cells[j-1,i-1]:=FloatToStr(v[i][j]);//,ffgeneral,1,5);

```

```

    end;
end;
for i:=1 to q do
for j:=1 to p do carpim[i][j]:=0;

for i:=1 to q do toplam[i]:=0;
dis:=0;
for i:=1 to q do
for j:=1 to p do
    dis:=dis+sqr(v[i][j]-v1[i][j]);
    dis:=sqr(dis);
if dis<0.000000000001 then
begin
    Timer1.Enabled:=False;
    showmessage('İşleminiz Bitmiştir.') ;
    for i:=1 to vs do maxu[i]:=-10;
    for i:=1 to vs do
begin
    for j:=1 to q do
    if maxu[i]<u1[i][j] then
begin
    maxu[i]:=u1[i][j];
    l[i]:=j;
end;
end;
end;
end;
end;
end;
end;
procedure TForm1.Panel2Click(Sender: TObject);
begin
    halt;
end;

```

```

procedure TForm1.Panel1Click(Sender: TObject);
begin
    vs:=20;
    for i:=1 to q do
    toplamm[i]:=0;
    for k:=1 to vs do
    for i:=1 to q do
    toplamm[i]:=toplamm[i]+(u[k][i]);
    for i:=1 to vs do
    for j:=1 to q do stt[i][j]:=0;
    for i:=1 to vs do
    for j:=1 to q do
    stt[i][j]:=sqr(veri[i][1]-v[j][1]);
    for i:=1 to q do
    st[i]:=0;

```

```

for i:=1 to q do
for k:=1 to vs do
st[i]:=st[i]+ u[k][i]*stt[k][i];
for i:=1 to q do
std[i]:=0;
for i:=1 to q do
std[i]:=sqrt(st[i]/toplamm[i]);
vs:=33;
for i:=1 to vs do
for j:=1 to q do
zz[i][j]:=0;
for i:=1 to vs do
for j:=1 to q do
zz[i][j]:=sqr(veri[i][1]-v[j][1]);
for i:=1 to vs do
for j:=1 to q do zzz[i][j]:=0;

for i:=1 to vs do
for j:=1 to q do
zzz[i][j]:=zz[i][j]/(2*sqr(std[j]));
for i:=1 to vs do
for j:=1 to q do
u[i][j]:=exp(-(zzz[i][j]));
for i:=1 to vs do tu[i]:=0;
for i:=1 to vs do
for j:=1 to q do tu[i]:= tu[i]+u[i][j];
for i:=1 to vs do
for j:=1 to q do g[i][j]:=0;

for i:=1 to vs do
for j:=1 to q do g[i][j]:=(u[i][j]/tu[i]);

vs:=20;
for i:=1 to vs do
for j:=1 to q*2 do w[i][j]:=0;

for i:=1 to vs do
begin
w[i][1]:=g[i][1];
w[i][2]:=g[i][1]*veri[i][1];
w[i][3]:=g[i][2];
w[i][4]:=g[i][2]*veri[i][1];
end;
for i:=1 to vs do
for j:=1 to q*2 do wt[j][i]:=0;

for i:=1 to vs do
for j:=1 to q*2 do wt[j][i]:=w[i][j];

```

```

for i:=1 to q*2 do
for j:=1 to vs do
for k:=1 to q*2 do wwt[i][k]:=0;

for i:=1 to q*2 do
for j:=1 to q*2 do
for k:=1 to vs do
wwt[i][j]:=wwt[i][j]+wt[i][k]*w[k][j];

for i:=1 to q*2 do
for j:=1 to vs do
for k:=1 to vs do wwt[i][k]:=0;
for i:=1 to q*2 do
for k:=1 to vs do
for j:=1 to q*2 do wwt[i][k]:=wwt[i][k]+ters[i][j]*wt[j][k];
for i:=1 to q*2 do gg[i]:=0;
for i:=1 to q*2 do
for j:=1 to vs do
gg[i]:=gg[i]+wwt[i][j]*veri[j][2];

b[1][1]:=gg[1]; b[1][2]:=gg[2];
b[2][1]:=gg[3]; b[2][2]:=gg[4];
for i:=1 to vs do yt[i]:=0;
for i:=1 to vs do
for j:=1 to q do yt[i]:=yt[i]+g[i][j]*(b[j][1]+b[j][2]*veri[i][1]);

for i:=1 to vs do mutlakfark[i]:=0;
for i:=1 to vs do mutlakfark[i]:=abs(x[i][2]-(yt[i]*(max[2]-
mn[2])+mn[2]))/x[i][2];;
ortfark:=0;
for i:=1 to vs do
ortfark:=ortfark+mutlakfark[i]*100;
ortfark:=ortfark/vs;
Edit1.Text:=FloatToStr(ortfark);

//Eğitim kümesi için hatanın hesaplanması bitti
//doğrulama ve test için üyelik değerlerinin bulunması
for i:=1 to vs do
for j:=1 to q do
begin
StringGrid1.Cells[3,i-1]:=IntToStr(i);
StringGrid1.Cells[j-1,i-1]:=FloatToStr(yt[i]*(max[2]-mn[2])+mn[2]);
end;

//DOĞRULAMA VERİSİ İÇİN HATA
for i:=21 to 28 do yt[i]:=0;

```

```

for i:=21 to 28 do
for j:=1 to q do yt[i]:=yt[i]+g[i][j]*(b[j][1]+b[j][2]*veri[i][1]);

for i:=21 to 28 do mutlakfark[i]:=0;

for i:=21 to 28 do
mutlakfark[i]:=abs(x[i][2]-(yt[i]*(max[2]-mn[2])+mn[2]))/x[i][2];

ortfark:=0;
for i:=21 to 28 do
ortfark:=ortfark+mutlakfark[i]*100;
ortfark:=ortfark/8;
Edit2.Text:=FloatToStr(ortfark);

for i:=29 to 32 do yt[i]:=0;
for i:=29 to 32 do
for j:=1 to q do
yt[i]:=yt[i]+g[i][j]*(b[j][1]+b[j][2]*veri[i][1]);

for i:=29 to 32 do
mutlakfark[i]:=0;

for i:=29 to 32 do
mutlakfark[i]:=abs(x[i][2]-(yt[i]*(max[2]-mn[2])+mn[2]))/x[i][2];
ortfark:=0;
for i:=29 to 32 do
ortfark:=ortfark+mutlakfark[i]*100;
ortfark:=ortfark/4;
Edit3.Text:=FloatToStr(ortfark);

//İleriye Dönük tahmin değerlerinin hesaplanması
Timer2.Enabled:=True;
k:=32;
end;
procedure TForm1.Panel3Click(Sender: TObject);
begin
Form1.WindowState:=wsminimized;
Form2.show;
Form2.WindowState:=wsmaximized;;
end;

procedure TForm1.Timer2Timer(Sender: TObject);
begin
k:=k+1;
showmessage(IntToStr(k));
veri[k][1]:=veri[k-1][2];
for j:=1 to q do zz[k][j]:=0;
for j:=1 to q do

```

```

zz[k][j]:=sqr(veri[k][1]-v[j][1]);

for j:=1 to q do zzz[k][j]:=0;
for j:=1 to q do
  zzz[k][j]:=zz[k][j]/(2*sqr(std[j]));
for j:=1 to q do
  u[k][j]:=exp(-(zzz[k][j]));
tu[k]:=0;
for j:=1 to q do tu[k]:= tu[k]+u[k][j];
for j:=1 to q do g[k][j]:=0;
for j:=1 to q do g[k][j]:=u[k][j]//tu[k];
for j:=1 to q do
yt[k]:=yt[k]+g[k][j]*(b[j][1]+b[j][2]*veri[k][1]);
veri[k][2]:=yt[k];
StringGrid3.Cells[0,k-32]:=FloatToStr(yt[k]*(max[2]-mn[2])+mn[2]);
if k>=40 then Timer2.Enabled:=False;

end;
end.

```

ÖZGEÇMİŞ**KİŞİSEL BİLGİLER**

Adı Soyadı :Nevin GÜLER
Doğum Yeri ve Tarihi :İzmir/ 26.07.1980

EĞİTİM VE AKADEMİK BİLGİLER

Lise : Beştepeler Lisesi 1994-1997
Lisans : 1997-2002 Muğla Üniversitesi, Fen-Edebiyat
Fakültesi, İstatistik ve Bilgisayar Bilimleri
Y. Lisans : 2003-2006 Muğla Üniversitesi, Fen Bilimler
Enstitüsü, İstatistik ve Bilgisayar Bilimleri ABD
Yabancı Dil : İngilizce

MESLEKİ BİLGİLER

2004-2006 : Muğla Üniversitesi, Fen-Edebiyat Fakültesi, Araştırma Görevlisi.