T.C.
TUNCELİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

SOSYAL TABANLI SEZGİSEL OPTİMİZASYON ALGORİTMALARIYLA SINIFLANDIRMA KURALLARININ KEŞFİ

YÜKSEK LİSANS TEZİ

Soner KIZILOLUK

Anabilim Dalı: Elektrik-Elektronik Mühendisliği

DANIŞMAN

Yrd. Doç. Dr. Bilal ALATAŞ

OCAK-2013

T.C.
TUNCELİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

SOSYAL TABANLI SEZGİSEL OPTİMİZASYON ALGORİTMALARIYLA SINIFLANDIRMA KURALLARININ KEŞFİ

YÜKSEK LİSANS TEZİ

Soner KIZILOLUK

(092103106)

Tezin Enstitüye Verildiği Tarih : 21 Ocak 2013
Tezin Savunulduğu Tarih : 14 Ocak 2013

Tez Danışmanı : Yrd. Doç. Dr. Bilal ALATAŞ (T.Ü)

Diğer Jüri Üyeleri : Yrd. Doç. Dr. Oktay GÖKTAŞ (T.Ü)

Yrd. Doç. Dr. Ömer ÇELİK (T.Ü)

OCAK-2013

Soner KIZILOLUK tarafından hazırlanan SOSYAL TABANLI SEZGİSEL OPTİMİZASYON ALGORİTMALARIYLA SINIFLANDIRMA KURALLARININ KEŞFİ adlı bu tezin Yüksek Lisans tezi olarak uygun olduğunu onaylarım.

Yrd. Doç. Dr. Bilal ALATAŞ

Tez Yöneticisi

Bu çalışma, jürimiz tarafından oy birliği/oy çokluğu ile Elektrik Elektronik Mühendisliği Anabilim Dalında Yüksek Lisans tezi olarak kabul edilmiştir. Bu tez, Tunceli Üniversitesi Fen Bilimleri Enstitüsü tez yazım kurallarına uygundur.

Başkan : Yrd. Doç. Dr. Oktay GÖKTAŞ (T.Ü)

Üye : Yrd. Doç. Dr. Bilal ALATAŞ (T.Ü)

Üye : Yrd. Doç. Dr. Ömer ÇELİK (T.Ü)

Tarih : 14 Ocak 2013

ÖNSÖZ

Çalışmalarım boyunca, değerli görüş ve katkılarıyla beni yönlendiren, her konuda desteğini esirgemeyen, kıymetli tecrübelerinden faydalandığım tez danışmanım Sayın Yrd.Doç.Dr. Bilal ALATAŞ'a teşekkürü borç bilirim. Ayrıca YLTUB011-14 kodlu proje ile çalışmalarıma maddi destek sağlayan Tunceli Üniversitesi Bilimsel Araştırma Projeleri Birimi'ne teşekkür ederim.

Soner KIZILOLUK
TUNCELİ-2013

İÇİNDEKİLER

ÖZET

Optimizasyon bir işi daha iyi yapma işlemidir. Örneğin optimizasyondaki bir f(x) fonksiyonunda sonuç değerlerini minimum yapacak x değerleri bulmak istenir. Optimizasyon problemlerini çözmek için değişik yöntemler önerilmiştir. Bu yöntemlerden bazılarında doğal süreçlerden esinlenilmiştir. Örnek olarak Karınca Koloni Optimizasyon Algoritması verilebilir. Bazı diğer yöntemlerde de sosyal olaylardan esinlenilmiştir. Sosyal tabanlı yöntemlerin sayısı fazla olmamakla birlikte en çok bilineni tabu arama algoritmasıdır. Son zamanlarda ise araştırmacılar öğretme-öğrenme tabanlı algoritma, Emperyalist Yarışmacı Algoritma ve Parlamenter Optimizasyon Algoritmasını geliştirmişlerdir.

Bu tez çalışmasında veri madenciliği, optimizasyon, sezgisel optimizasyon, emperyalist yarışmacı algoritma ve parlamenter optimizasyon algoritması hakkında genel bilgi verilmiştir. Visual C# programında, parlamenter optimizasyon algoritmasına uygun program yazılmıştır. UCI veri ambarından alınan 4 farklı veri tabanı bu programda uygulanmış ve sınıflandırma kuralları elde edilmiştir. Ayrıca elde edilen sonuçlar da WEKA programında elde edilen sonuçlar ile karşılaştırılmıştır.

Anahtar Kelimeler: Parlamenter Optimizasyon Algoritması, Veri Madenciliği, Sınıflandırma Kural Keşfi

ABSTRACT

Optimization is the process of making something better. For example, in an f(x) function in optimization, it is asked for finding x values which make the outcome of the f(x) minimum. Different methods have been proposed for solving optimization problems. Some of these processes have been inspired by natural processes. Ant Colony Optimization Algorithm can be given as an example. Some of the other methods have been inspired by social events. Although the number of social based methods is limited, the most known one is tabu search algorithm. Recently, researchers have developed teaching-learning based algorithm, Imperialist Competitive Algorithm, and Parliamentary Optimization Algorithm.

In this thesis study, general information about data mining, optimization, heuristic optimization, imperialist competitive algorithm and parliamentary optimization algorithm is given. A program code compatible to the parliamentary optimization algorithm was written in Visual C#. Four different databases obtained from UCI data warehouse were applied onto this program and classification rules were obtained. Furthermore, the results were compared with the results obtained from WEKA program.

Key Words: Parliamentary Optimization Algorithm, Data Mining, Classification Rule Mining

ŞEKİLLER LİSTESİ

Sayfa No

TABLOLAR LİSTESİ

Sayfa No

KISALTMALAR

EYA : Emperyalist Yarışmacı Algoritma
GA : Genetik Algoritma
POA : Parlamenter Optimizasyon Algoritması
PSO : Parçacık Sürü Optimizasyonu

SEMBOLLER LİSTESİ

a	: Alt sınır
b	: Bayrak
$c1,c2,c3, c4$	: Ağırlıklar
C_n	: Emperyalistin maliyeti
c_n	: Normalize edilmiş maliyet
$Cost$	: Ülkenin maliyeti
DN	: Doğru negatif
DP	: Doğru pozitif
$f(p_n)$	: Aday üyelerin uygunluk fonksiyonu
L	: Birey sayısı
M	: Grup sayısı
$N.C._n$	: n. İmparatorluğun başlangıç koloni sayısı
$N.T.C._n$	: Normalize edilmiş toplam maliyet
N_{col}	: Tüm koloilerin sayısı
$ort(Q^i)$	: Aday üyelerin ortalaması
$ort(R^i)$	: Asil üyelerin ortalaması
$Ö_i max, Ö_i min$	: Veritabanındaki ilgili özelliğin, maksimum ve minimum değeri
p'	: Asil üyenin yönelme sonrası yeni değeri
p_0	: Asil üyenin yönelme öncesi değeri
P_n	: Emperyalistin normalize edilmiş gücü
p_n	: Bireyler
$power^i$	: i. grubun gücü
Q^i	: Adaylar vektörü
R^i	: Asiller vektörü
$T.C._n$	: İmparatorluğun toplam maliyeti
$ü$	: Üst sınır
X	: Koloninin emperyaliste gelişigüzel hareket değeri
YN	: Yanlış negatif
YP	: Yanlış pozitif
θ	: Gelişigüzel sapma değeri
ξ	: 1'den küçük pozitif bir değer
π	: 0.5, 2 arasında gelişigüzel bir değer

1.GİRİŞ

Farklı alanlardaki araştırmacılar doğanın çözdüğü birçok zor problemden esinlenerek birçok fikir elde etmişlerdir. Bu alanlardan biri de optimizasyondur. Doğanın seçtiği evrimsel yaklaşımların avantajları genetik algoritmalarda ve türevlerinde optimizasyon için önerilmiştir. Hayvanların davranışları parçacık sürü algoritmaları (Kennedy ve Eberhart, 1995) ve karınca koloni algoritmalarıyla (Dorigo vd., 1991) sonuçlanmıştır. Son zamanlarda ise insanların sosyal davranışlarını simüle etme mühendislikte bazı çözümlerde yol göstermiştir. Bunun sonucunda Emperyalist Yarışmacı Algoritma (EYA) (Atashpaz-Gargari ve Lucas, 2007) ve Parlamenter Optimizasyon Algoritması (POA) (Borji, 2007) gibi sosyal tabanlı algoritmalar ortaya çıkmıştır.

Günümüzde bilişim teknolojileri baş döndürücü bir hızda gelişmektedir. Bu gelişme beraberinde bir sorunu da getirmiştir. Bilişim sistemleri sayesinde artık tüm veriler sayısal ortamda saklanmaktadır. Bilişim teknolojisi tüm bu verileri saklamaya yeterli olabilir. Fakat bu veriler ne işe yarayacaktır? Bu verilerden bazı avantajlar kazanabilecek miyiz? Biriken veriler gerçek anlamda "bilgiye" dönüştürülebilecek midir? Bu tür sorulara veri madenciliği ile olumlu yanıt vermek mümkündür. Basit bir tanım yapmak gerekirse, veri madenciliği, büyük ölçekli veriler arasından değeri olan bir bilgiyi elde etme işidir (Özkan, 2008).

Veri madenciliğinde kullanılan modeller tahmin edici ve tanımlayıcı olmak üzere ikiye ayrılır. Tahmin edici modellerde; sonuçları bilinen verilerden bir model geliştirilir ve bu modelden yararlanılarak bilinmeyen veri kümeleri için sonuç tahmini amaçlanır. Tanımlayıcı modellerde ise karar vermeye rehberlik etmede kullanılabilecek mevcut verilerdeki örüntülerin tanımlanması sağlanmaktadır (Tiryaki, 2006).

Veri madenciliği modellerini gördükleri işleve göre üç başlık altında incelemek mümkündür:

- Sınıflandırma ve regresyon
- Kümeleme
- Birliktelik kuralları

Sınıflandırma ve regresyon tahmin edici model iken, kümeleme ve birliktelik kuralları tanımlayıcı modellerdir (Delice, 2008; Akpınar, 2000).

Veri madenciliği modellerinden biri olan sınıflandırma modeli, sosyal tabanlı algoritmalardan olan POA ile birlikte kullanılarak sınıflandırma kural keşfi bu tez çalışmasının temelini oluşturmaktadır. Bölüm 2'de veri madenciliği hakkında genel bilgi verilecektir. Bölüm 3'te optimizasyon ve sezgisel optimizasyon hakkında genel bilgi verilecektir. Bölüm 4'te ise sırasıyla emperyalist yarışmacı algoritmasından ve parlamenter optimizasyon algoritmasından bahsedilecektir. Bölüm 5'te POA ile Visual C#'da yazılan programda uygulanan veri tabanları hakkında genel bilgi verilecektir. Bununla birlikte verilerin POA'da nasıl uygulandığı anlatılacak, POA ve WEKA programlarından elde edilen sonuçlar değerlendirilecektir.

2. VERİ MADENCİLİĞİ

Bilişim alanındaki gelişmeler ve bilişim sistemlerinin yaşamımızın birçok alanına girmesiyle beraber yaptığımız her işlem sayısal ortamda kaydedilmektedir. Örneğin internette, bankada, hastanede veya markette yaptığımız her işlem artık veri tabanlarında tutulmaya başlanmıştır. Bunun sonucunda elde edilen veriler sayılamayacak kadar büyüktür. Elimizdeki büyük veriler arasında faydalı ve işimize yarayacak bilgiler bulunabilir. Toprak örtüsü altındaki değerli madenleri gün yüzüne çıkarmak istenir ve bu da madencilikle sağlanır. Benzer şekilde büyük veriler arasındaki yararlı bilgilere de ulaşılmak istenir. Bu durum "veri madenciliği" kavramının ortaya çıkmasına sebep olmuştur.

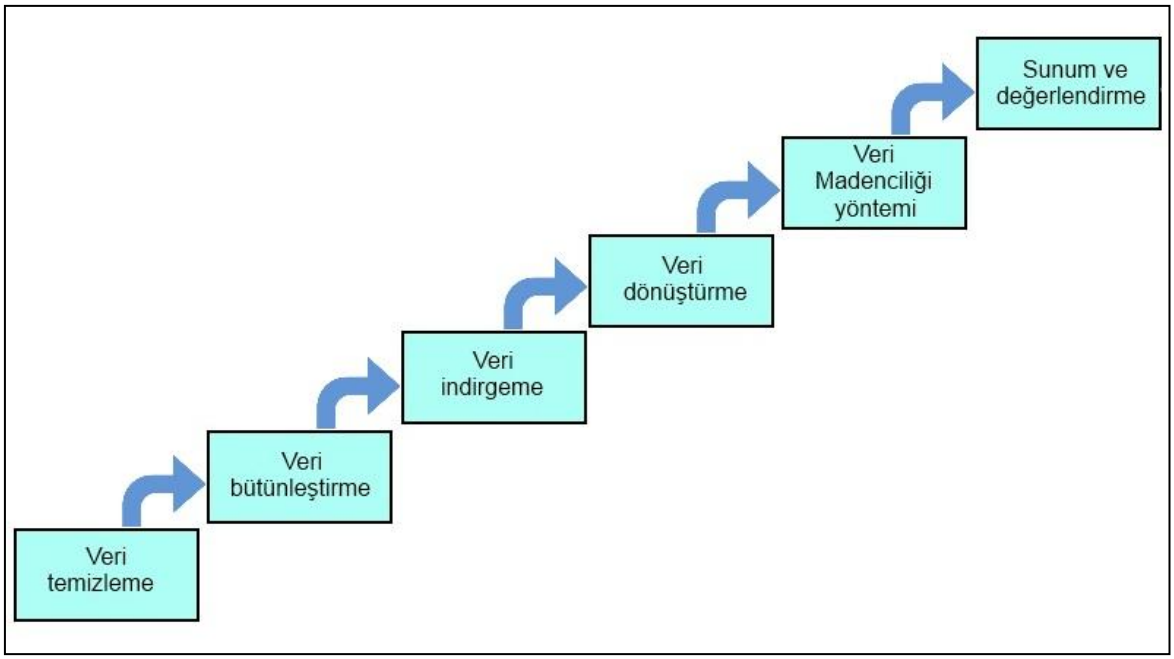
Veri madenciliği konusunda çeşitli tanımlamalar yapılabilir. Veri madenciliği, büyük ölçekli veriler arasından "değeri olan" bir bilgiyi elde etme işlemidir. Başka bir tanıma göre veri madenciliği, bir kurumda üretilen tüm verilerin belirli yöntemler kullanarak, var olan bilgiye veya gelecekte ortaya çıkabilecek gizli bilgilere ulaşma işlemidir (Özkan, 2008).

2.1. Veri Madenciliği Süreci

Veri madenciliği bir süreç olarak değerlendirilmelidir. Bu süreç 6 adımdan oluşur.

1. Veri temizleme
2. Veri bütünleştirme
3. Veri indirgeme
4. Veri dönüştürme
5. Veri madenciliği algoritmasını uygulama
6. Sonuçları sunum ve değerlendirme

Bu adımlar Şekil 2.1'de gösterilmiştir.



Şekil 2.1. Veri madenciliği sürecindeki adımlar (Özkan, 2008)

2.1.1. Veri Temizleme

Bazı uygulamalarda, üzerinde işlem yapılacak veriler eksik veya uygun olmayan değerlere sahip olabilir. Veri tabanındaki bu tutarsız veriler gürültü olarak adlandırılır. Bu gibi durumlarda bu verilerin temizlenmesi gerekecektir. Bunun için aşağıdaki yöntemler kullanılabilir (Özkan, 2008).

- Eksik değer içeren kayıt veritabanından silinebilir.
- Kayıp değerler yerine genel bir sabit kullanılabilir.
- Diğer tüm verilerdeki değerlerin ortalaması eksik değer yerine yazılabilir.
- Diğer tüm veriler yerine sadece bir sınıfa ait verilerdeki değerlerin ortalaması eksik değer yerine yazılabilir.
- Verilere uygun bir tahmin yapılarak eksik değer yerine kullanılabilir.

2.1.2. Veri Bütünleştirme

Farklı veritabanlarından alınan verilerin birlikte değerlendirilmeye alınabilmesi için farklı türdeki verilerin tek türe dönüştürülmesi yani bütünleştirilmesi gerekecektir (Özkan, 2008).

2.1.3. Veri İndirgeme

Veri madenciliği uygulamalarında eğer çözümlemeden elde edilecek sonucun değişmeyeceğine inanılıyorsa veri sayısı yada değişkenlerin sayısı azaltılabilir. Veri indirgeme çeşitli biçimlerde yapılabilir (Özkan, 2008):

- Veri birleştirme veya veri küpü: Çözümlemeler sadece belirli boyutlara göre yapılabilir.
- Boyut indirgeme: Veriler arasında bir seçme işlemi yapılarak, gereksiz veriler veri tabanından silinebilir böylece boyut azaltılması sağlanabilir.
- Veri sıkıştırma: Büyük veri kümeleri sıkıştırılarak daha az yer kaplamaları sağlanabilir.
- Örnekleme: Büyük veri toplulukları yerine onları temsil edecek daha küçük veri toplulukları kullanılabilir.
- Genelleme: Verilerin tek tek değil genel kavramlarla ifade edilmesidir.

2.1.4. Veri Dönüştürme

Bazı durumlarda verileri, veri madenciliği çözümlemelerine aynen katmak uygun olmayabilir. Bu yüzden kullanılan verinin kullandığımız veri madenciliği tekniğinde kullanılabilecek hale dönüştürülmesi gerekir (Özkan, 2008).

2.1.5. Veri Madenciliği Algoritmasını Uygulama

Veriyi hazır hale getirmek için yukarıdaki adımlardan uygun görülenleri yapılır. Veri hazır hale getirildikten sonra veri madenciliği algoritmaları kullanılabilir (Özkan, 2008).

2.1.6. Sonuçlar Sunum ve Değerlendirme

Veri madenciliği algoritması veriler üzerinde uygulandıktan sonra sonuçlar elde edilir ve bu sonuçlar değerlendirilir.

2.2. Veri Madenciliği Modelleri

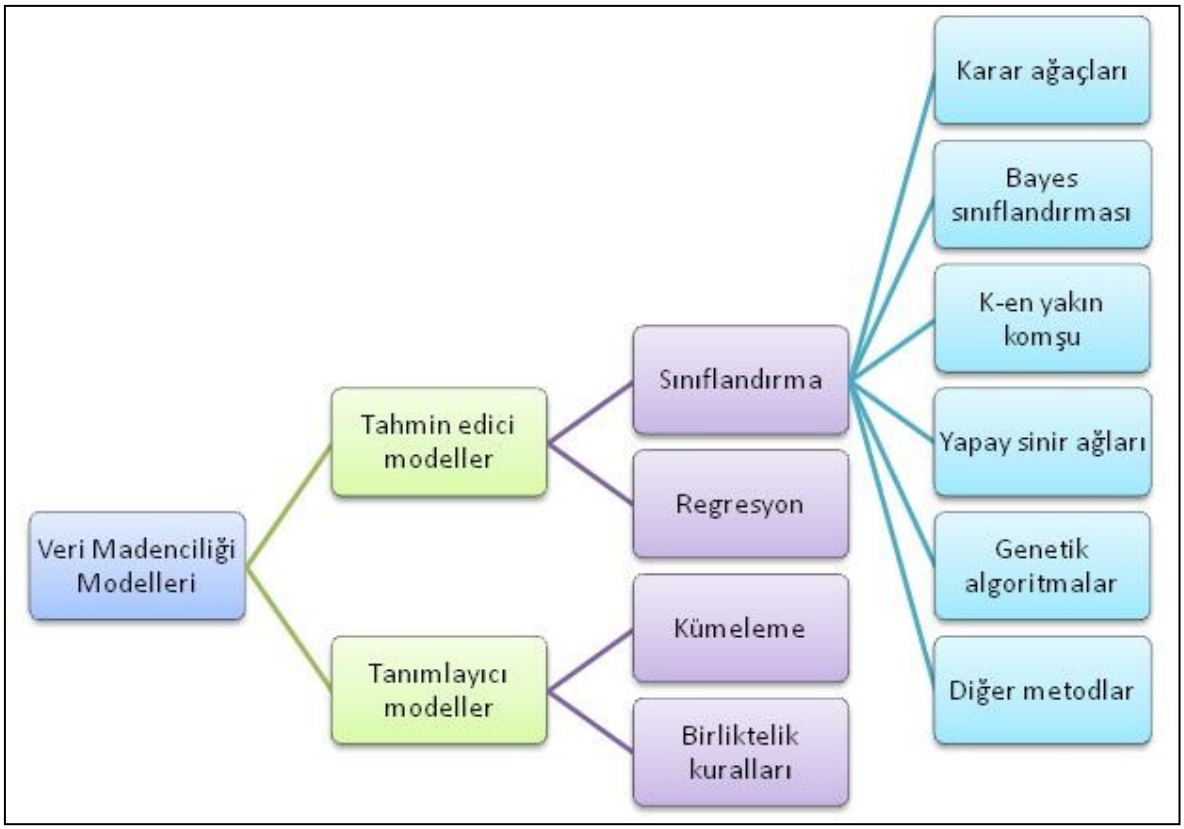
Veri madenciliğinde kullanılan modeller, tahmin edici modeller ve tanımlayıcı modeller olmak üzere iki ana başlık altında toplanabilir.

Tahmin edici modellerde; sonuçları bilinen verilerden çıkarak bir model geliştirilir ve bu modelden yararlanarak sonuçları bilinmeyen veri kümeleri için sonuç değerlerinin tahmin edilmesi amaçlanır. Örneğin bir bankanın kredilerle ilgili veritabanında müşterinin özellikleri ve krediyi geri ödeyip ödemediği bilgisini tutan veriler bulunur. Bu verilere uygun olarak kurulan modelle, daha sonra kredi talep eden müşterinin özelliklerine göre müşterinin krediyi geri ödeyip ödemeyeceğinin tahmini yapılabilir. Tanımlayıcı modellerde ise karar vermeye rehberlik etmede kullanılabilecek mevcut verilerdeki örüntülerin tanımlanması sağlanmaktadır (Tiryaki, 2006).

Veri madenciliği modellerini gördükleri işlevlere göre üçe ayırmak mümkündür (Delice, 2008; Akpınar, 2000):

- Sınıflandırma ve Regresyon
- Kümeleme
- Birliktelik Kuralları

Veri madenciliğindeki modeller ve modellere göre gördükleri işlevleri Şekil 2.2' de gösterilmiştir.



Şekil 2.2. Veri madenciliği modelleri

2.2.1. Sınıflandırma ve Regresyon

Sınıflandırma ve regresyon tahmin edici modellerdir. Sınıflandırma kategorik değerlerin, regresyon ise süreklilik gösteren değerlerin tahmininde kullanılır (Demirel, 2010).

Sınıflandırma ve regresyon geleceğe ait veri tahmini yapmak ve önemli veri sınıflarını açıklayan modeller çıkarmak için kullanılır. Sınıflandırma modeli iki adımdan oluşur. İlk adımda sınıflandırma algoritması kullanılarak o sınıfa ait kurallar bulunur. İkinci adımda ise, sınıfı bilinmeyen farklı ve yeni veriler geldiğinde, bu verilerin ilk adımda elde edilen kurallara göre uygun sınıf ataması yapılır. Veri madenciliğinde genel olarak, sınıf etiketlerinin belirlenmesinde kullanılan tahmin işlemi sınıflandırmadır. Regresyon ise sürekli değerlerin belirlenmesinde kullanılan tahmin işlemidir (Delice, 2008; Han ve Kamber, 2001).

2.2.2. Kümeleme

Kümeleme verilerin kendi aralarındaki benzerliklerin göz önüne alınarak gruplandırılmasıdır. Bu nedenle pek çok alanda kullanılabilir. Pazar araştırmalarında, resim işleme, desen tanımlama ve uzaysal harita verilerinin analizlerinde kullanılmaktadır (Özkan, 2008). Kümeleme analizi, sınıflandırma ve regresyondan farklı olarak, bilinen sınıf etiketini dikkate almaksızın veri nesnelerini analiz etmektedir. Kümeler çeşitli etiketler oluşturmak için kullanılır. Aynı küme içindeki nesnelerin özellikleri birbirine benzerdir. Farklı kümelerdeki nesnelerin benzerlikleri ise çok düşüktür (Delice, 2008; Han ve Kamber, 2001).

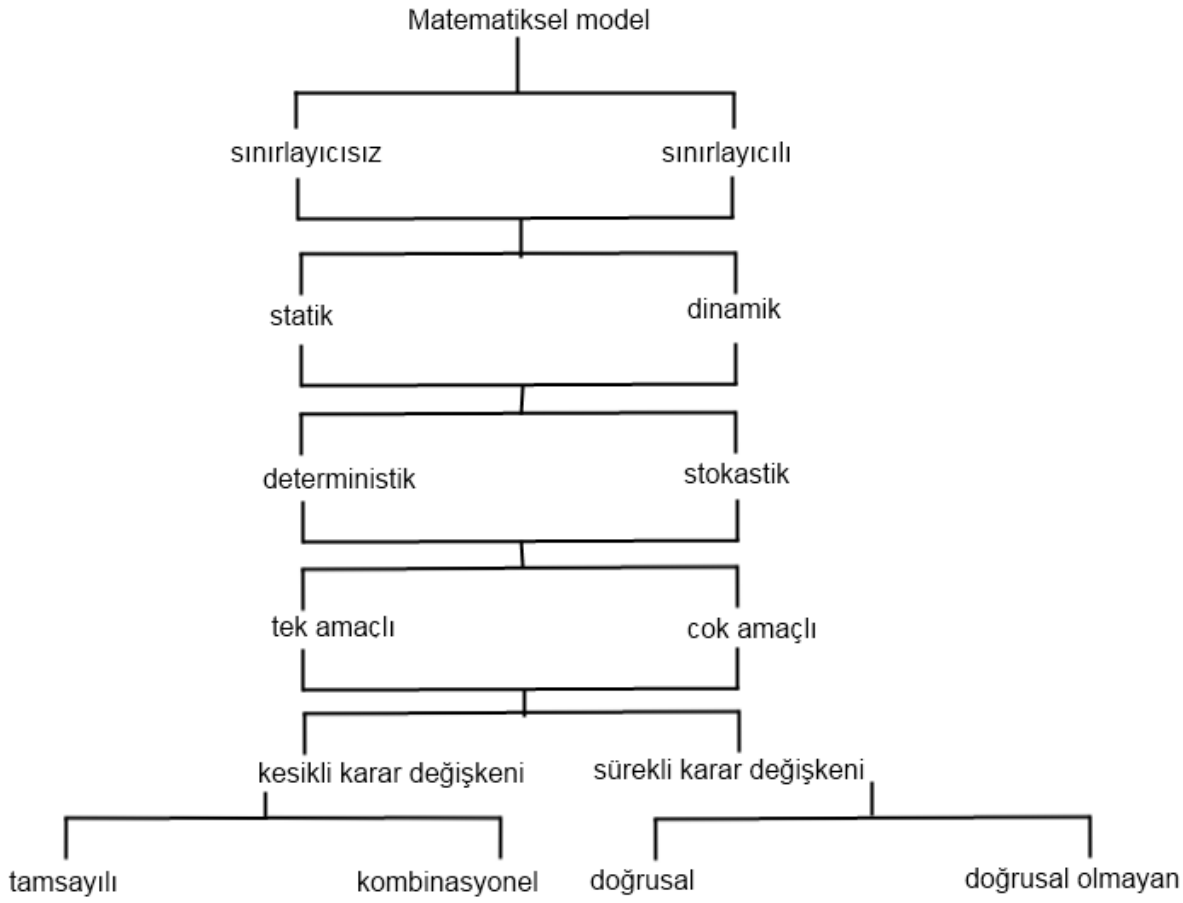
2.2.3. Birliktelik Kuralları

Veritabanı içinde yer alan kayıtların birbirleriyle ilişkileri incelenerek, hangi olayların eş zamanlı olarak birlikte gerçekleşebileceklerini ortaya koymaya çalışan veri madenciliği yöntemleri bulunmaktadır. Bu ilişkilerin belirlenmesi sonucunda birliktelik kuralları elde edilir.

Birliktelik kuralları özellikle pazar araştırmalarında kullanılır. Pazar sepet analizi adı verilen uygulamalarda bu yöntemler kullanılmaktadır. Bu tür çözümlemelerin amacı, müşterinin alışveriş alışkanlıklarını ortaya koymaktır. Bir müşteri herhangi bir ürün aldığında, sepetine başka hangi ürünü koyduğu belirli bir olasılığa göre hesaplanır. Birlikte satılan ürünler belirlendiğinde, mağazalarda raflar ona göre düzenlenir ve müşterinin o ürünlere daha kolay ulaşması sağlanmış olur (Özkan, 2008).

3. OPTİMİZASYON

Optimizasyon bir problemin en iyi çözümünü elde etme işlemidir. Optimizasyonun performansını etkileyen ve kontrolümüz altında değerleri olan değişkenlere *karar değişkenleri* denir. Karar değişkenlerinin amaç üzerindeki etkilerinin analitik olarak gösterilmesiyle *amaç fonksiyonu* oluşturulur. Çoğu durumda, karar değişkenlerinin sadece belirli değerleri kullanılmalıdır. Karar değişkenlerinin değerleri üzerindeki bu sınırlandırmalara *sınırlayıcılar* denir. O halde farklı bir ifadeyle optimizasyon, karar değişkenlerinin mümkün olan tüm kombinasyonları arasından verilen tüm sınırlayıcıları sağlayan ve amaç fonksiyonunu en iyi hale getiren (maksimizasyon ya da minimizasyon) kombinasyonun bulunması işidir (Alataş, 2007).



Şekil 3.1. Optimizasyon için matematiksel modeller

Şekil 3.1'de optimizasyon için matematiksel modeller görülmektedir. Eğer karar değişkenleri üzerinde hiçbir sınırlama yoksa sınırlayıcısız, en azından bir sınırlama olması durumunda sınırlayıcılı olur. Eğer problem tek bir dönem için çözülecekse statik model, birden fazla dönem göz önüne alınarak çözülecekse dinamik model kullanılır. Modelin algoritmada işletilmesi esnasında belirli, kesin parametre veya girdiler kullanılıyorsa model deterministik, olasılık özelliği varsa model stokastiktir. Eğer birden fazla amaç varsa, problemler çok amaçlıdır. Eğer tüm karar değişkenleri pozitif reel (gerçel) değerler alıyorsa sürekli optimizasyon problemi söz konusudur. Tüm karar değişkenlerinin tamsayı değerler alması gerekiyorsa kesikli optimizasyon problemi ortaya çıkar (Alataş, 2007).

Optimizasyon algoritmalarının çoğu, sistemin modeli ve amaç fonksiyonu için matematiksel modellere ihtiyaç duymaktadır. Karmaşık sistemler için matematiksel modelin kurulması çoğu zaman zordur. Model kurulsa bile, çözüm zamanı maliyeti çok yüksek olduğundan kullanılamamaktadır. Klasik optimizasyon algoritmaları, büyük ölçekli kombinasyonsal ve doğrusal olmayan problemlerde yetersizdir. Bu tür algoritmalar, verilen bir probleme bir çözüm algoritması uyarlamada etkin değildir. Bu da çoğu durumda, geçerliliğinin onaylanması zor olabilen bazı varsayımları gerektirir. Genellikle klasik algoritmaların doğal çözüm mekanizmalarından dolayı, *ilgilenilen problem algoritmanın onu idare edeceği şekilde modellenir*. Klasik optimizasyon algoritmalarının çözüm stratejisi genellikle amaç ve sınırlayıcıların tipine (doğrusal, doğrusal olmayan vb.) ve problemi modellemede kullanılan değişkenlerin tipine (tamsayı, reel) bağlıdır. Bunların etkinliliği aynı zamanda problem modellemede çözüm uzayı (konveks, konveks olmayan vb.), karar değişken sayısı ve sınırlayıcı sayısına oldukça bağlıdır. Diğer önemli bir eksiklik ise farklı tipte karar değişkenleri, amaç ve sınırlayıcıların olması durumunda problem formülasyonlarına uygulanabilecek genel çözüm stratejileri sunmamalarıdır. Yani çoğu algoritma belirli tipteki amaç fonksiyonu ya da sınırlayıcıların olduğu modelleri çözmektedir. Ancak çoğu yönetim bilimi, bilgisayar, mühendislik gibi bir çok farklı alandaki optimizasyon problemleri eşzamanlı olarak formülasyonlarında farklı tipteki karar değişkenleri, amaç fonksiyonu ve sınırlayıcıları gerektirir. Bu yüzden sezgisel optimizasyon algoritmaları önerilmiştir. Bunlar son yıllarda oldukça popüler yöntemler haline gelmiştir çünkü bunların hesaplama gücü iyidir ve dönüşümleri kolaydır. Yani tek amaç fonksiyonlu bir problem için yazılmış bir sezgisel program, kolaylıkla çok amaçlı bir probleme ya da farklı bir probleme uyarlanabilmektedir (Alataş, 2007).

3.1. Sezgisel Optimizasyon

Gerçek yaşam problemlerinin çoğunda problemin çözüm uzayı sonsuz veya tüm çözümlerin değerlendirilemeyeceği kadar büyük olur. Bunun için kabul edilebilir bir sürede çözümlerin değerlendirilerek iyi bir çözümün bulunması gerekmektedir. Böyle problemler için kabul edilebilir bir sürede çözümlerin değerlendirilmesiyle aslında tüm çözüm uzayında "bazı çözümlerin" değerlendirilmesi aynı anlama gelmektedir. Bazı çözümlerin neye göre ve nasıl seçileceği sezgisel tekniğe göre değişir. Maalesef değerlendirmeye dahil olan çözümlerin içerisinde optimal çözümün yer alması garanti edilememektedir. Bu sebeple de sezgisel tekniklerin bir optimizasyon problemine önerdiği çözüm, optimal değil iyi çözüm olarak algılanmalıdır (Cura, 2008).

Sezgisel algoritmalara gerek duyulmasının sebepleri aşağıdaki gibidir.

a) Optimizasyon problemi kesin çözümü bulma işleminin tanımlanamadığı bir yapıya sahip olabilir.

b) Anlaşılırlık açısından sezgisel algoritmalar karar verici açısından çok daha basit olabilir.

c) Sezgisel algoritmalar, öğrenme amaçlı ve kesin çözümü bulma işleminin bir parçası olarak kullanılabilir.

d) Matematik formülleriyle yapılan tanımlamalarda genellikle gerçek dünya problemlerinin en zor tarafları (hangi amaçlar ve hangi sınırlamalar kullanılmalı, hangi alternatifler test edilmeli, problem verisi nasıl toplanmalı) ihmal edilir. Model parametrelerini belirleme aşamasında kullanılan verinin hatalı olması, sezgisel yaklaşımın üretebileceği alt optimal çözümden daha büyük hatalara sebep olabilir.

Bir problem için geliştirilmiş bir sezgisel algoritma, aşağıdaki faktörler göz önüne alınarak değerlendirilebilmektedir.

• Çözüm Kalitesi ve Hesaplama Zamanı: Çözüm kalitesi ve hesaplama zamanı bir algoritmanın etkinliğinin değerlendirilmesi için önemli kriterlerdir. Bundan dolayı bir algoritma, ayarlanabilir parametreler setine sahip olmalı ve bu parametreler kullanıcıya önemlilik açısında hesaplama maliyeti ile çözüm kalitesi arasında bir vurgulamanın

yapılabilmesine imkân vermelidir. Diğer bir deyişle, çözüm kalitesi ile hesap zamanı arasındaki ilişki kontrol edilebilmelidir.

• Kod Basitliği ve Gerçeklenebilirlik: Algoritma prensipleri basit olmalı ve genel olarak uygulanabilir olmalıdır. Bu durum problem yapısı ile ilgili başlangıçta çok az bilgiye sahip olunması halinde bile algoritmanın yeni alanlara kolaylıkla uygulanabilmesini sağlar.

• Esneklik: Algoritmalar modelde, sınırlamalarda ve amaç fonksiyonlarında yapılacak değişiklikleri kolayca karşılayabilmelidir.

• Dinçlik: Yöntem, başlangıç çözümünün seçimine bağlı olmaksızın her zaman yüksek kaliteli, kabul edilebilir çözümleri üretebilme kabiliyetine sahip olmalıdır.

• Basitlik ve Analiz Edilebilirlik: Karmaşık algoritmalar, esneklik ve çözüm kalitesi açısından basit algoritmalardan daha zor analiz edilebilmektedir. Algoritma kolayca analiz edilebilir olmalıdır.

• Etkileşimli Hesaplama ve Teknoloji Değişimleri: Algoritma içinde insan-makine etkileşimini kullanma fikri çoğu sistemde yaygın olarak gerçekleştirilmektedir. Herkesçe bilindiği gibi iyi bir kullanıcı ara yüzü herhangi bir bilgisayar sistemini veya algoritmayı daha çekici yapmaktadır. Bunun en önemli avantajı çözümlerin grafiksel olarak sergilenebilmesidir (Karaboğa, 2004).

Genel amaçlı sezgisel yöntemler; biyolojik tabanlı, fizik tabanlı, sosyal tabanlı, müzik tabanlı ve kimya tabanlı olmak üzere çeşitli gruplara ayrılmaktadır. Ayrıca bunların birleşimi olan melez yöntemler de vardır. Genetik Algoritma (GA) (Holland, 1975), diferansiyel gelişim algoritması (Storn ve Price, 1995) ve karınca koloni algoritmaları (Dorigo vd., 1991) biyolojik tabanlı, ısıl işlem algoritması (Kirkpatrick vd., 1983) ve elektromanyetizma algoritması (Birbil ve Fang, 2003) fizik tabanlı, tabu arama (Glover, 1989) sosyal tabanlı, yapay kimyasal reaksiyon optimizasyon algoritması (Alataş, 2011) kimya tabanlı ve armoni arama algoritması (Geem, 2001) müzik tabanlı algoritma ve modellerdir.

Algoritma tek bir çözümden başlayıp bunu operatörlerle ilerletiyorsa bunlara tek noktalı yöntemler denir ve tabu arama, ısıl işlem gibi bütün yerel arama tabanlı sezgisel algoritmalar bu gruba girer. Çok noktadan yani bir popülasyon üzerinden çözüme başlanıp bu farklı noktalarla optimizasyon yapılıyorsa bunlar çok noktalı ya da popülasyon tabanlı

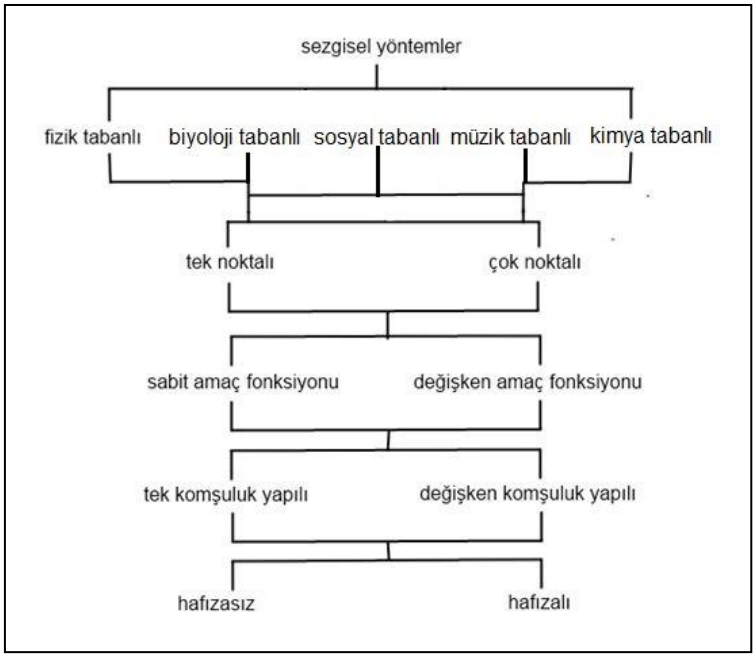
yöntemlerdir. Bunlara GA, Parçacık Sürü Optimizasyonu (PSO), karınca koloni algoritmaları, arı koloni algoritmaları, yapay bağışıklık sistemleri ve elektromanyetizma algoritması örnek olarak verilebilir.

Bazı sezgisel algoritmalar problemin gösterimini gerçekleştirirken amaç fonksiyonunu sabit tutar ve sabit amaç fonksiyonlu olarak adlandırılırlar. Bazıları da örneğin rehberli yerel arama algoritmasındaki gibi değiştirir ve değişen amaç fonksiyonlu olarak adlandırılırlar. Değiştirmekteki amaç yerel minimumdan kurtulmaktır. Bu mantık, yerel minimumdan kaçmak için bazen diğer sezgisel algoritmalara da uygulanabilmektedir.

Çoğunlukla sezgisel yöntemler tek bir komşuluk yapısında çalışır ve tek komşuluk yapılı olarak sınıflandırılabilir. Bazıları da değişken komşuluk arama algoritmasında olduğu gibi arama işlemini sistematik bir şekilde değiştirerek birden fazla yerel arama yöntemiyle diğer çözüm alanlarına ulaşmaya çalışır ve değişken komşuluk yapılı şeklinde sınıflandırılabilir.

Algoritmalar çalışırken daha önceki durumlar ya da en iyi durumlar hatırlanıyorsa hafızalı, hatırlanmıyorsa hafızasız şeklinde sınıflandırılabilir. Örneğin PSO ve tabu arama hafızalı, GA hafızasızdır (Alataş, 2007).

Tüm bu sezgisel yöntemler Şekil 3.2'de görülmektedir.



Şekil 3.2. Sezgisel yöntemler

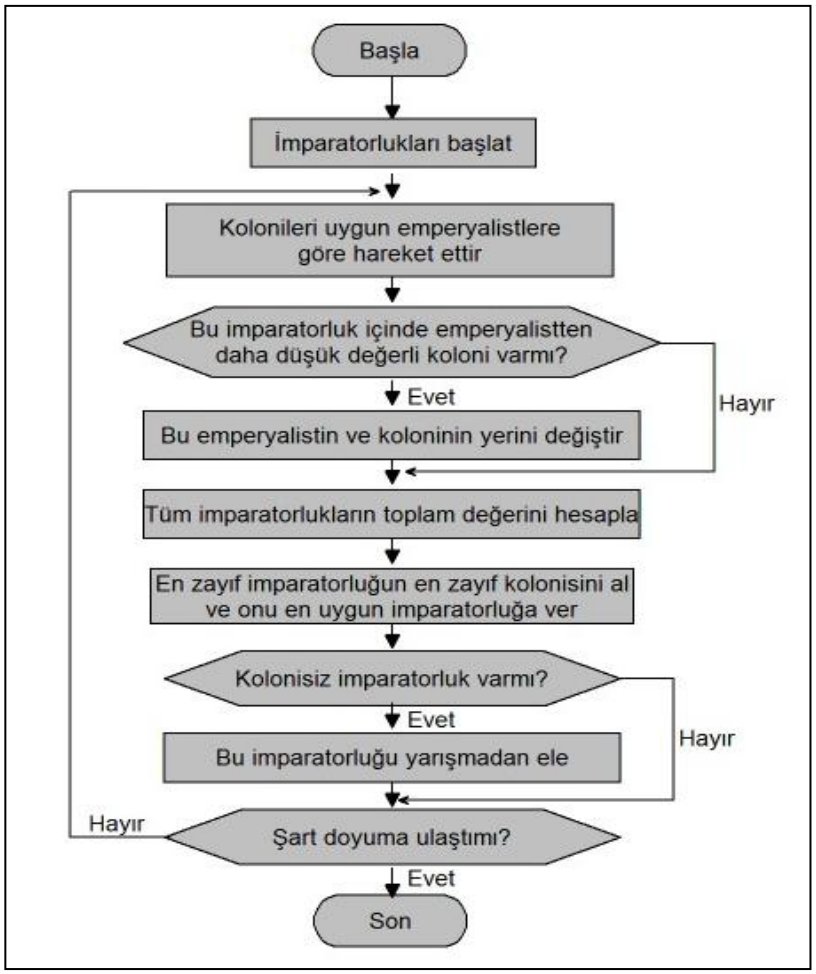
4. SOSYAL TABANLI GÜNCEL SEZGİSEL OPTİMİZASYON ALGORİTMALARI

Literatürde dört tane sosyal tabanlı sezgisel optimizasyon algoritması bulunmaktadır. Bunlardan en bilineni ve uygulaması en çok yapılanı tabu arama algoritmasıdır. Yakın zamanda ise üç tane yeni sosyal tabanlı algoritma daha önerilmiştir: EYA (Atashpaz-Gargari ve Lucas, 2007), POA (Borji, 2007) ve öğretme-öğrenme tabanlı algoritma (Rao vd., 2012). Henüz yeni önerildiğinden, bu yöntemlerle ilgili çok az sayıda çalışma bulunmaktadır.

Bu bölümde ise EYA ve POA hakkında genel bilgi verilecektir. Ayrıca POA kullanılarak, dört farklı veri tabanında sınıflandırma kural keşfi uygulaması yapılacak ve elde edilen sonuçlar yazılacaktır.

4.1. Emperyalist Yarışmacı Algoritma

Diğer evrimsel algoritmalar gibi EYA'da bir başlangıç popülasyonu ile başlar (dünyadaki ülkeler). Popülasyondaki birkaç en iyi ülke emperyalist olmak için seçilir ve kalanlar da bu emperyalistlerin kolonisi olur. Tüm koloniler bu emperyalist devletler arasında dağıtılır. Tüm kolonilerin dağıtımından sonra, bu koloniler uygun emperyalistlere doğru hareket etmeye başlar. Bir imparatorluğun toplam gücü emperyalistin ve onun kolonilerinin gücüne bağlıdır. Daha sonra tüm imparatorluklar arasında emperyalistik yarış başlar. Eğer bir imparatorluk bu yarışta gücünü arttıramaz ve başarılı olamazsa yarıştan elenir. Bu yarışta güçlü imparatorluklar gücüne güç katarken zayıf imparatorlukların ise gücü azalır ve sonunda zayıf imparatorluklar yıkılır. Bu yarış en son tek bir imparatorluk kalana kadar devam eder ve sonunda diğer tüm ülkeler bu imparatorluğun bir kolonisi olur. Bu ideal yeni dünyada koloniler emperyalist ile aynı konuma ve güce sahip olacaktır. Şekil 4.1' de algoritmanın akış şeması görülmektedir (Atashpaz-Gargari ve Lucas, 2007).



Şekil 4.1. Emperyalist yarışmacı algoritma akış şeması

Emperyalist yarışmacı algoritma güncel bir yöntem olduğu için, bu yöntem ile yapılan çalışma sayısı da oldukça azdır. Analog devre optimizasyonu (Razzaghpour ve Rusu, 2011), birliktelik kurallarının keşfi (Khademolghorani, 2011), gezici robotun global konumlandırılması (Tamimi vd., 2010), kablosuz sensör ağının yerleştirilmesi (Sayadnavard vd., 2010), çevrimiçi PI kontrolü (Movahed ve Yazdani, 2011), DC motorun hız kontrolü (Ghalehpardaz ve Shafiee, 2011), uyarlanabilir emperyalist yarışmacı algoritma (Abdechiri vd., 2010) ve kaotik temelli yapay sinir ağı öğrenme (Abdechiri vd., 2010) EYA ile yapılmış bazı çalışmalardır.

4.1.1. Başlangıç İmparatorluklarını Üretme

Optimizasyonun amacı problem değişkenlerinden en optimal çözümü bulmaktır. Değer değişkenleri için bir dizi oluşturulur. Genetik algoritmada buna "kromozom" adı verilirken emperyalist yarışmacı algoritmada "ülke (country)" adı verilir. Bir N_{var} boyutlu

optimizasyon probleminde bir ülke bir $1x\ N_{var}$ dizisidir. Bu dizi Denklem 4.1'deki gibi tanımlanır.

$$country = [u_1, u_2, \ldots, u_{Nvar}] \tag{4.1}$$

Ülkedeki değişken değerleri noktalı kayan sayılarla temsil edilir. Bir ülkenin maliyeti (Cost) $u_1, u_2, \ldots, u_{Nvar}$ değişkenlerindeki değer fonksiyonu ile ölçülür.

$$Cost = f(country) = f(u_1, u_2, \ldots, u_{Nvar}) \tag{4.2}$$

Optimizasyon algoritmasına başlamak için N_{pop} boyutundaki başlangıç popülasyonu oluşturulur. İmparatorlukları oluşturmak için N_{imp} sayıdaki en güçlü ülkeler seçilir. Geri kalan N_{col} ise bir imparatorluğa ait olan kolonilerdir. Bu şekilde emperyalist ve koloni olmak üzere 2 tip ülkeye sahip olunur.

Başlangıç imparatorluklarını oluşturmak için koloniler, emperyalistler arasında emperyalistlerin güçlerine göre dağıtılır. Böylece emperyalistlerin başlangıç koloni sayısı gücü ile doğru orantılı olur. Kolonileri emperyalistler arasında doğru orantılı dağıtmak için, bir emperyalistin normalize edilmiş maliyeti Denklem 4.3'te gösterildiği gibi tanımlanır:

$$C_n = c_n - maks_i\{ci\} \tag{4.3}$$

C_n, n. emperyalistin maliyeti ve c_n ise normalize edilmiş maliyetidir. Tüm emperyalistlerin normalize edilmiş maliyetlerini içeren, her emperyalistin normalize edilmiş gücü ise Denklem 4.4'te gösterildiği gibi tanımlanır:

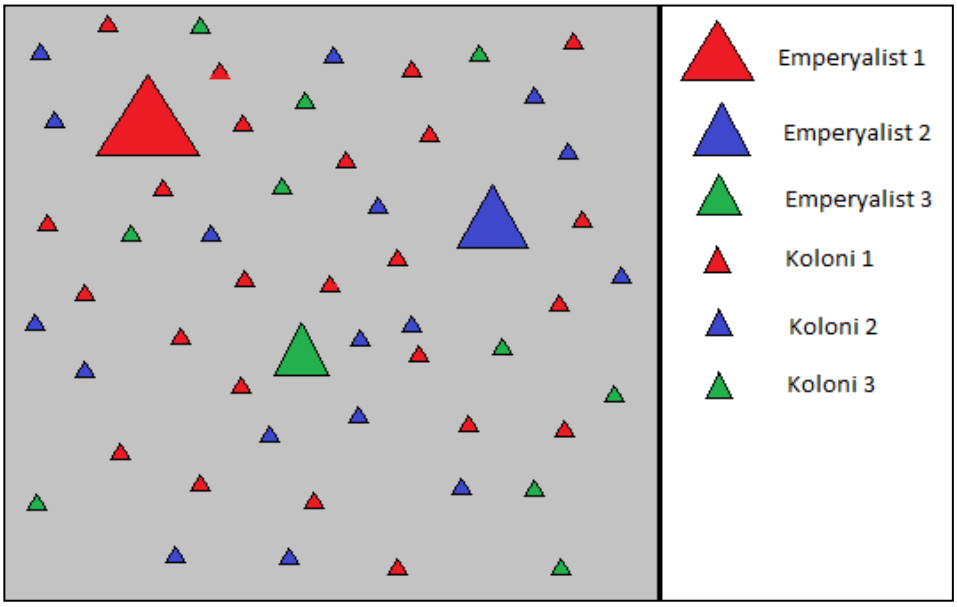
$$p_n = \left| \frac{C_n}{\sum_{i=1}^{N_{imp}} C_i} \right| \tag{4.4}$$

Bir başka görüşe göre, bir emperyalistin normalize edilmiş gücü, emperyalistin sahip olduğu koloniler kısmıdır. Böylece bir imparatorluğun kolonilerinin başlangıç sayısı Denklem 4.5'teki gibi olacaktır.

$$N.C._n = round\{p_n . N_{col}\} \tag{4.5}$$

$N.C._n$ n. İmparatorluğun başlangıç koloni sayısıdır. N_{col} tüm kolonilerin sayısıdır. Kolonileri emperyalistlere dağıtmak için gelişigüzel $N.C._n$ koloni seçilir ve emperyaliste

verilir. Bu koloniler emperyalist ile birlikte *n.* imparatorluğu oluşturur. Şekil 4.2 her imparatorluğun ilk popülasyonunu gösterir ve daha güçlü imparatorlukların daha çok koloniye sahip olduğu görülebilir (Abdechiri vd., 2010).



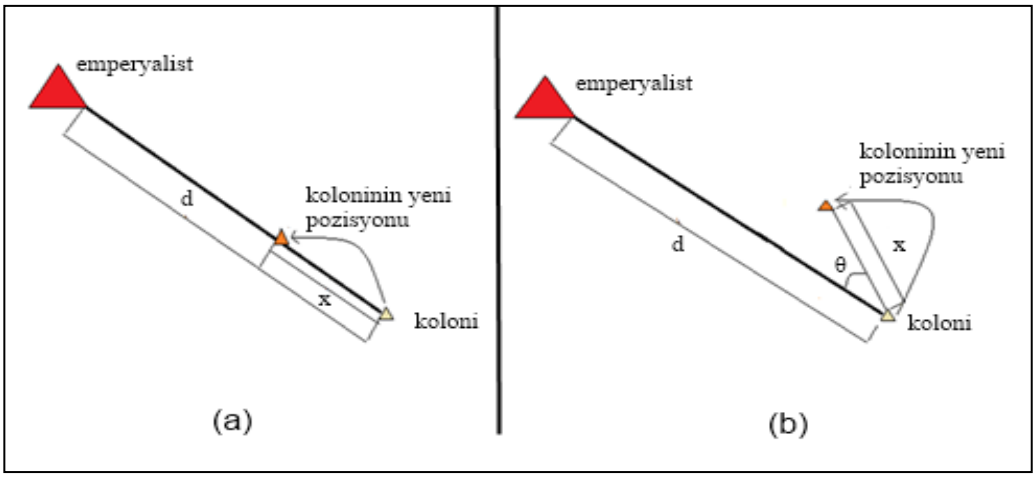
Şekil 4.2. İmparatorluğun ilk popülasyonu

4.1.2. Kolonilerin Hareketi

Zamanla emperyalist ülkeler kolonilerini arttırmaya başlar. Bu durum kolonilerin emperyaliste doğru hareketi şeklinde gerçekleşir. Şekil 4.3a'da kolonin *x* birim emperyaliste hareketi gösterilmektedir. Hareketin yönü koloniden emperyaliste doğru bir vektördür. Şekildeki *x* gelişigüzel bir değerdir.

$$X \sim U\,(0\,,\beta\ \times d\,) \tag{4.6}$$

β 1 den büyük bir sayı ve *d* ise aradaki uzaklıktır. $\beta > 1$ olması koloninin emperyaliste yaklaşmasına neden olur.



Şekil 4.3. a) Koloninin emperyaliste hareketi b) Koloninin yeni pozisyonu

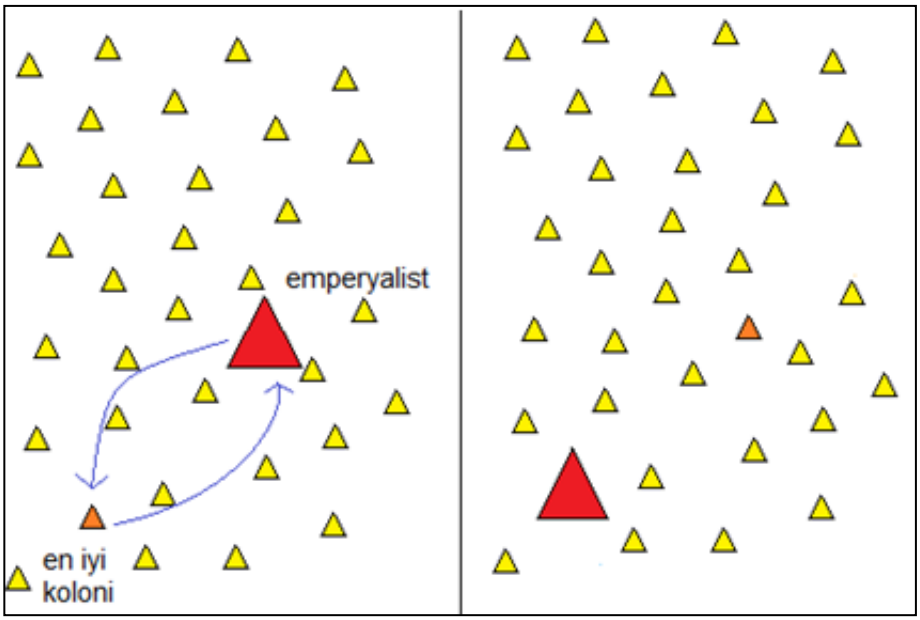
Emperyalistin etrafında farklı noktalar aramak için hareketin yönüne gelişigüzel bir değerde sapma eklenir. Şekil 4.3b'de θ gelişigüzel bir değerdir.

$$\theta \sim U\,(-\gamma, \gamma) \tag{4.7}$$

γ Orijinal yönden sapmanın değerini ayarlayan parametredir. Bununla beraber β ve γ değerleri keyfidir. (Atashpaz-Gargari ve Lucas, 2007).

4.1.3. Emperyalist Ve Koloninin Yerini Değiştirme

Bir koloni emperyaliste doğru ilerlerken, emperyalistten daha iyi bir konuma erişebilir. Böyle bir durumda emperyalist ile koloni yer değiştirir. Daha sonra algoritma, emperyalistin yeni pozisyonu ve kolonilerin bu yeni pozisyona hareketi ile devam eder. Şekil 4.4, koloni ile emperyalistin yer değiştirmesini gösterir. İmparatorluğun en iyi kolonisi koyu renk ile gösterilmiştir. Bu koloni emperyalistten daha düşük maliyete sahiptir. Şekil 4.4'te imparatorluğun, emperyalist ile koloninin yer değiştirmesinden sonraki durumu da gösterilmiştir (Atashpaz-Gargari ve Lucas, 2007).



Şekil 4.4. Koloni ile emperyalistin yer değiştirmesi

4.1.4. İmparatorluğun Toplam Gücü

Bir imparatorluğun toplam gücü çoğunlukla emperyalist ülkenin gücünden etkilenir. Fakat imparatorluktaki kolonilerin gücü göz ardı edilebilir olsa da bu imparatorluğun toplam gücü üstünde bir etkiye sahiptir. Bu durum Denklem 4.8'de gösterildiği gibi toplam maliyeti tanımlayarak modellenmiştir:

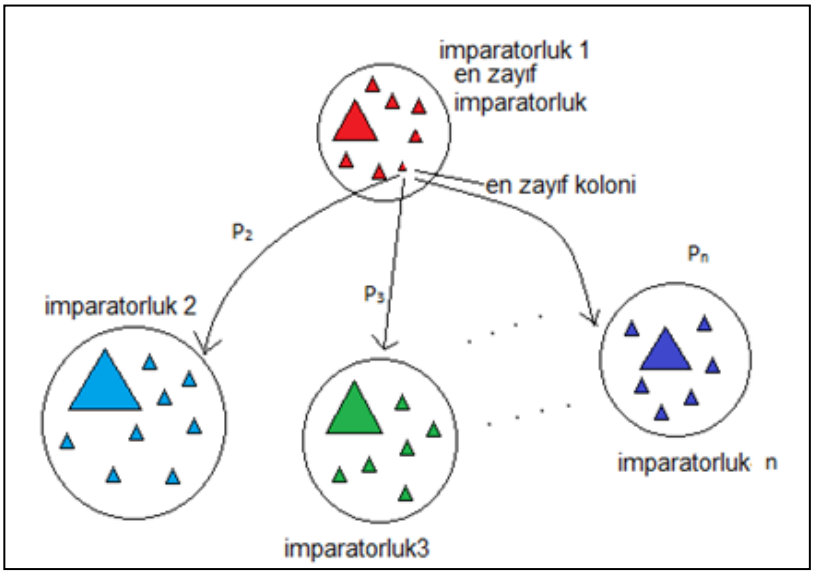
$$T.C._n = Cost(imperialist_n) + \xi\ mean\{Cost(colonies\ of\ empire_n)\} \quad (4.8)$$

n. imparatorluğun toplam maliyeti $T.C._n$ 'dir ve ξ 1' den küçük olan pozitif bir sayıdır. ξ için küçük bir değer sadece emperyalist tarafından belirlenen imparatorluğun gücünü etkiler ve bu değeri arttırmak imparatorluğun toplam gücüne karar vermede kolonilerin rolünü arttırır. 0,1 değeri birçok uygulamada ξ için kullanılmıştır (Atashpaz-Gargari ve Lucas, 2007).

4.1.5. Emperyalistik Yarış

Bütün imparatorluklar diğer imparatorlukların kolonilerini ele geçirmeye çalışır ve bunları kontrol eder. Bu emperyalistik yarış gitgide zayıf imparatorlukların gücünde azalmayı ve daha güçlü olanların gücünde artışı beraberinde getirir. Bu yarış, zayıf imparatorlukların zayıf kolonilerinden bazılarının alınması ve diğer imparatorlukların bu kolonileri ele geçirmeye çalışmaları şeklinde modellenmiştir. Şekil 4.5, modellenmiş

emperyalistik yarışın büyük bir resmini gösterir. Bu yarışta imparatorlukların her biri zayıf kolonileri ele geçirme ihtimaline sahiptir. Fakat güçlü imparatorlukların bu kolonileri ele geçirme olasılıkları daha fazla olacaktır.



Şekil 4.5. Emperyalistik yarışma

Yarışa başlamak için ilk olarak her imparatorluğun toplam gücüne göre ele geçirme olasılığı bulunmalıdır.

$$N.T.C._n = T.C._n - max_i\{T.C._i\} \tag{4.9}$$

Denklem 4.9'daki $T.C._n$ *n.* imparatorluğun toplam maliyeti ve $N.T.C._n$ normalize edilmiş toplam maliyetidir. Ele geçirme olasılıkları ise Denklem 4.10'daki gibidir.

$$p_{p_n} = \left| \frac{N.T.C._n}{\sum_{i=1}^{N_{imp}} N.T.C._i} \right| \tag{4.10}$$

Bahsi geçen kolonileri imparatorlukların sahip olduğu ele geçirme olasılıkları arasında dağıtmak için, Denklem 4.11'de gösterilen *P* vektörü oluşturulur.

$$P = [p_{p_1}, p_{p_2}, \dots, p_{p_{N_{imp}}}] \tag{4.11}$$

Sonra *P* ile aynı boyutta gelişigüzel sayılardan oluşan Denklem 4.12'deki *R* vektörü oluşturulur.

$$R=[r_1,r_2,\ldots,r_{Nimp}] \quad , \quad r_1,r_2,\ldots,r_{Nimp} \sim U(0,1) \tag{4.12}$$

Daha sonra *P*'den *R*'yi çıkarma işlemi Denklem 4.13'de gösterilmiştir:

$$D=P-R=[D_1,D_2,\ldots D_{Nimp}]=[P_{P1}-r_1\ ,\ P_{P2}-r_2\ ,\ldots,\ P_{Pnimp}-r_{nimp}] \tag{4.13}$$

Vektör *D*'ye dayanarak, *D*'nin maksimum indeksi ile alakalı olan imparatorluğun kolonileri elde edilir (Atashpaz-Gargari ve Lucas, 2007).

4.1.6. Bir Noktada Birleşme

Güçsüz imparatorluklar emperyalistik yarışta geride kalır ve çöker. Onun kolonileri ise diğer imparatorluklar arasında paylaştırılır. En güçlü imparatorluk dışındaki bütün imparatorluklar çöktükten sonra tüm koloniler tek bir imparatorluğun kontrolüne girer. Bu ideal yeni dünyada aynı pozisyon ve maliyette olan koloniler, onlarla aynı pozisyon ve maliyete sahip olan bir emperyalist tarafından kontrol edilecektir. Böyle bir durumda emperyalist yarışa son verilmeli ve algoritma durdurulmalıdır (Atashpaz-Gargari ve Lucas, 2007).

4.2. Parlamenter Optimizasyon Algoritması

İnsan sosyal hayatının birçok durumunda rekabetçi davranışlar gözlemlenebilmektedir. POA'da genetik algoritmalar ve parçacık sürü algoritmaları gibi olasılıksal, iteratif ve popülasyon tabanlı global optimizasyon tekniğidir. Bilhassa bu metot, parlamentonun kontrolünü ele geçirme çalışmaları sırasındaki grup içi ve gruplar arası çekişmeleri simüle etmeye çalışmıştır.

Parlamentarizm olarak da bilinen parlamenter sistem, yasaları yapma ve düzenleme gücüne sahip olan hükümet sistemidir. Parlamento üyeleri genel seçimlerde halk tarafından seçilmiştir. İnsanlar genellikle favori partisine oy verirler. Parlamento üyeleri politik partilere üyedirler. Onlar parlamento seçimlerinde partilerini desteklerler. Parlamento üye grupları, ait oldukları partiyi temel alarak, partiler arasındaki yarışmayı diğer partiler üzerinde üstünlük kazanma şeklinde sonuçlandırmak için çalışırlar. Hemen hemen tüm demokratik ülkelerde, politik partiler parlamento popülasyonunu oluştururlar (Borji, 2007; Borji ve Hamidi, 2009).

Parlamento seçimlerinde genelde iki sistem vardır: Çoğunluk seçim sistemi ve orantılı temsil sistemi. Çoğunluk seçim sisteminde her seçim bölgesinden yalnız bir üye seçilir. Orantılı temsil sisteminde bir seçim bölgesinden birkaç üye seçilebilir. Genelde her politik parti aday listelerini sunar ve seçmenler oylayacağı politik parti listesini seçebilir. Partilere aldığı oylarla orantılı olarak parlamentoda sandalye verilir.

Parlamento içindeki veya dışındaki politik partiler, değişik seviyedeki güçte üyelere sahiptir. Partideki bu insanlar az güç ile diğer asil üyeler üzerinde iyi bir etki bırakmak için uğraşırlar. Bunu onların destekleri ve seçimler sırasındaki oyları için yaparlar. Partinin önemli üyeleri yarışlarda devreye girer ve asil üyeler arasında destek bulmaya çalışır. Diğer bir yandan asil üyeler daha becerikli kişilere eğilimlidir ve genelde inandıkları kişilere oy verirler. Bu süreçte, yüksek kapasiteli genel üyeler önceki adaylarla yerleri değiştirilir. Bu yarışma parti içindeki bireyler arasında olur. Diğer bir yarış ise partiler arasındadır. Partiler daha fazla güç elde etmek için yarışırlar. Partilerin başarı için iki temel amacı vardır: Parlamentodaki en yüksek sayıdaki sandalyeye sahip olmak ve hükümetin kontrolünü almak (Borji, 2007; Borji ve Hamidi, 2009).

POA ise gerçek hayattaki parlamento seçimlerini simüle etmektedir. Algoritmadaki optimizasyon işlemi, ilk olarak birey popülasyonun oluşturulmasıyla başlar. Bu bireyler parlamentonun üyeleri olarak kabul edilir. Bir sonraki adımda, popülasyon bazı politik gruplar arasında dağıtılır ve yüksek uygunluktaki sabit sayıdaki üyeler grup adayı olarak seçilir.

Popülasyonun bölüştürülmesinden sonra, grup içi yarış başlar. Grup içi yarışmada asil üyeler kendilerine uygun adaylara doğru yönelir. Bu durum asil üye adaylarının vektörlerinin ağırlıklı ortalaması olarak modellenmiştir. Parti içi yarıştan sonra birkaç yüksek uygunluktaki aday grubun son adayları olarak kabul edilir. Bir sonraki adımda bu adaylar diğer grupların adayları ile yarışır. Bir gruptaki adayların ve asil üyelerin her ikisi de grubun toplam gücünün belirlenmesinde önemlidir. Adayların ana gücünün doğrusal kombinasyonu ve asil üyelerin ana gücü, bir grubun toplam uygunluğu olarak nitelendirilir.

Grup içi yarışmadan sonra gruplar arası yarış başlar. Politik gruplar parlamentoda kendi adaylarını kabul ettirmek için diğer gruplarla parlamento yarışı yapar. Bu metotta grubun rolü, bir adayı tanıttıktan sonra hâlâ korunur. Uygunluğu göz ardı edilen grupların gücü aşama aşama azalır ve sonunda çöker. Diğer bir yandan, güçlü gruplar adım adım daha

güçlü olmaya başlar ve yarışı kazanmak için daha fazla şans elde eder. Güçlü gruplar bazen birleşmek için anlaşırlar ve kazanma şanslarını arttırırlar. Gruptaki asil üyeler kendi gruplarındaki adaylarla beraber, kendileriyle benzer özellikte ve daha güçlü bir gruba katılırlar. Bu adımlar parlamentoda tek bir grupta birleşilene kadar devam eder. Tüm gruplar birleştikten sonra, asil üyeler lider olan adayla neredeyse eşit güce sahip olurlar. Algoritmanın akış şeması Şekil 4.6'da gösterilmiştir (Borji, 2007; Borji ve Hamidi, 2009).

Şekil 4.6. POA akış şeması

4.2.1. Popülasyonun Başlatılması

N_{var} boyutundaki başlangıç çözüm popülasyonu, gelişigüzel pozisyonlardaki d boyutlu problem uzayına yayılırlar. Popülasyonun her bireyi boyutsal sürekli vektör olarak kodlanmıştır.

$$P = [p_1, p_2, \dots, p_n], p_i \in IR \quad (4.14)$$

Verilen gruptaki bireylerden her biri asil üye ya da aday üye olacaktır. (Borji ve Hamidi, 2009).

4.2.2. Popülasyonun Bölümlendirilmesi

Başlangıç gruplarını oluşturmak için, popülasyon L sayıda bireyden oluşan M tane gruba bölünür.

$$N_{var} = M \times L \quad (4.15)$$

N_{var}, M ve L pozitif tam sayılar olmak üzere N_{var} Denklem 4.15'teki gibidir.

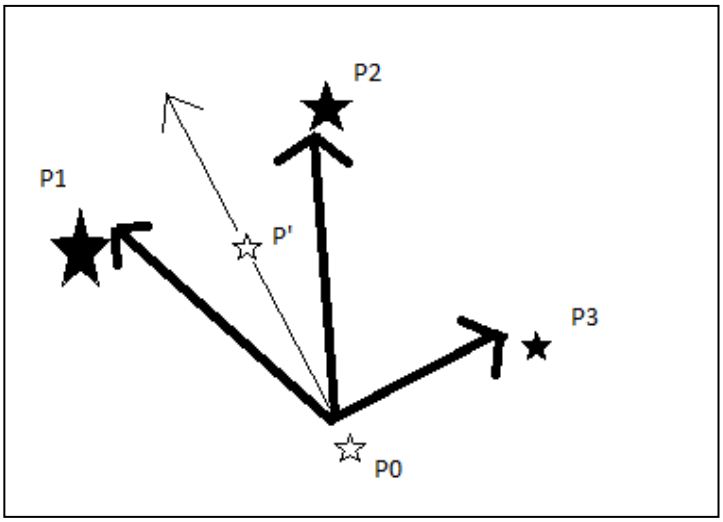
Yüksek uygunluktaki $\theta < L/3$ aday, her grubun adayı olarak nitelendirilir. Bu noktada tüm gruplar eşit sayıda üyeye sahip olurlar, fakat algoritmanın çalışması esnasında gruplar birleşme ve çökme mekanizmasından dolayı farklı sayıda birey elde edebilirler (Borji ve Hamidi, 2009).

4.2.3. Grup İçi Yarışma

Gruptaki asil üyeler, adaylar ve asil üyeler arasındaki yer alma etkileşiminden sonra, adaylara doğru yönelirler. Bu yönelme, bir üyeyi adaylara bağlayan vektörlerin ağırlıklı ortalamaları ile doğru orantılıdır. Her aday Denklem 4.16'da gösterildiği gibi kendi aday uygunluklarını arttırmak için ağırlıklandırılmıştır.

$$p' = p_0 + \Lambda \left(\frac{(p_1 - p_0).f(p_1) + (p_2 - p_0).f(p_2) + (p_3 - p_0).f(p_3)}{f(p_1) + f(p_2) + f(p_3)} \right) \quad (4.16)$$

Denklem 4.16'da p_0 asil üyenin yönelme öncesindeki değeridir. p_1, p_2 ve p_3 aday üyelerin değerleridir. $f(p_1)$, $f(p_2)$ ve $f(p_3)$ ise aday üyelerin uygunluk fonksiyonudur. p' ise asil üyenin aday üyelere doğru yönelmesi sonucundaki aldığı yeni değerdir. Λ 0,5 ile 2 arasında gelişigüzel bir sayıdır ve algoritmanın yerel arama alanı çevresindeki adayları aramasına olanak verir. Diğer alternatif bir mekanizmada ise ilk iterasyondan itibaren Λ değeri adım adım azaltılır. Şekil 4.7, yönlenme mekanizmasını gösterir (Borji ve Hamidi, 2009).



Şekil 4.7. Yönlenme mekanizması

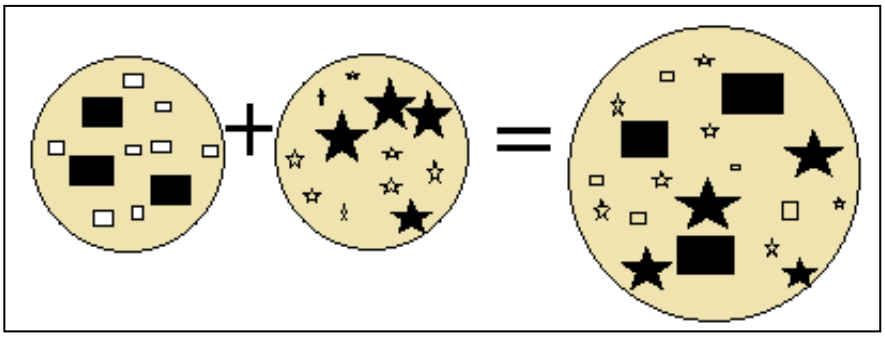
Asil bir üyenin değişmesine izin verilmesi, sadece üye büyük uygunluk değeri aldığında olur. Yönelmeden sonra asil üyeler, aday üyelerden daha yüksek uygunluk değerine sahip olabilir. Bu gibi durumlarda, adayların yer değişikliği gerçekleşir. $Q^i=[Q_1, Q_2, ..., Q_\theta]$ adaylar vektörü olsun ve $R^i=[R_{\theta+1}, R_{\theta+2}, ..., R_i]$ de *i.* gruptaki geri kalan asil üyeler olsun. Bu grubun gücü Denklem 4.17'de gösterildiği gibi hesaplanır (Borji ve Hamidi, 2009).

$$power^i = \frac{rn1\ x\ ort(Q^i) + rn2\ x\ ort(R^i)}{rn1+rn2}\ ; rn1 > rn2 \qquad (4.17)$$

Denklem 4.17'de $ort(Q^i)$ ve $ort(R^i)$ sırasıyla gruptaki aday üyelerin ve asil üyelerin değerlerinin ortalamasıdır. *rn1* ve *rn2* ise gelişigüzel değerlerdir. $power^i$ ise i. grubun gücünü ifade eder.

4.2.4. Gruplar Arası Yarışma

Güçlü gruplar bazen, kendi güçlerini arttırmak için bir gruba katılır ve birleşirler. Birleşmeyi gerçekleştirmek için gelişigüzel bir sayı üretilir ve bu sayı p_m'den küçük ise, λ sayıda en güçlü grup seçilir ve bir grupta birleştirilir. Şekil 4.8, bu birleşmeyi göstermektedir. Devam eden algoritma boyunca; zayıf gruplar, hesaplanan gücü korumak ve değer fonksiyonunu azaltmak için silinir. Birleştirmedeki gibi, gelişigüzel bir sayı üretilir ve sayı p_d 'den küçükse, *y* sayıda minimum güce sahip gruplar elenir (Borji ve Hamidi, 2009).



Şekil 4.8. Grupların birleşmesi

4.2.5. Durumun Sonlandırılması

Algoritma sonunda, bir grup yarışı kazanır ve onun en iyi elemanı optimizasyon probleminin çözümü olarak nitelendirilir. İki sonlandırma durumu mevcuttur. Maksimum sayıda iterasyona ulaşıldığında veya bazı başarılı iterasyon sonucunda uygunluk değerinde dikkate değer iyileşme gözlenmezse algoritma sonlandırılır (Borji ve Hamidi, 2009).

5. PARLAMENTER OPTİMİZASYON ALGORİTMASI KULLANILARAK SINIFLANDIRMA KURAL KEŞFİ UYGULAMASI

Bu tez çalışmasında, POA'ya uygun olarak Visual C#'da yazılmış bir program geliştirilmiştir. Program, UCI veri ambarında bulunan diabet (URL-1, 2012), Ecoli (URL-2, 2012) ve BUPA Liver Disorders (URL-3, 2012) ve New Thyroid (URL-4, 2012) olmak üzere dört farklı veritabanı üzerinde uygulanarak sınıflandırma kural keşfi yapılmıştır.

Bölüm 2.1.'de bahsedildiği gibi veri madenciliği süreci altı adımdan oluşmaktadır. Beşinci adım olan "veri madenciliği algoritması uygulama"sına geçmeden önceki adımlardan gerekli olan uygulanıp veriler algoritma için hazır hale getirilir. Bu çalışmada ise kullanılan veriler algoritma için hazır halde olduğundan dolayı ilk dört adım uygulanmamıştır ve veritabanları üzerinde hiç bir değişiklik yapılmadan programda kullanılmıştır.

Bu bölümde, WEKA veri madenciliği yazılımından, WEKA içinde bulunan ve bu çalışmada karşılaştırma amaçlı kullanılan sınıflandırma algoritmalarından, kullanılan her veri tabanının özelliklerinden, kural temsilinin nasıl yapıldığından bahsedilecektir. Ayrıca POA'nın uygulanmasıyla elde edilen sonuçlar ve bu sonuçların, WEKA veri madenciliği yazılımındaki bazı sınıflandırma algoritmalarının uygulanmasında elde edilen sonuçlar ile karşılaştırılması da yapılacaktır.

5.1. WEKA

WEKA, Yeni Zelanda Waikato üniversitesi tarafından geliştirilen ücretsiz bir veri madenciliği programıdır (URL5, 2012). Bu programda birçok sınıflandırma, kümeleme ve birliktelik algoritması mevcuttur. Bu çalışmada ise WEKA içinde mevcut olan sınıflandırma algoritmalarından, Jrip, Ridor, Part ve One-R algoritmaları, bizim bulduğumuz sonuçlar ile karşılaştırmak için kullanılmıştır.

5.1.1. Jrip Algoritması

Algoritma, tamamı pozitif örneklerden oluşan bir kural seti kurar ve gürültülü büyük veri setleri üzerinde etkili performans sergiler.

Bir kural oluşturmadan önce, öğrenme örneklerinin mevcut setini, geliştirilen set ve budanan set olmak üzere iki alt sete böler. Kural, geliştirilen set üzerindeki örneklerden yapılandırılır. Kural seti, boş bir kural kümesi ile başlar ve kurallar artırımlı olarak negatif örnekler arındırılıncaya kadar kural setine eklenir. Geliştirilen setten bir kural üretildikten sonra, budanan örnekler üzerinde kural setinin performansını geliştirmek için koşul, kuraldan silinir (Turna, 2011).

5.1.2. Ridor Algoritması

Algoritma ilk olarak bir varsayılan kural üretir. Daha sonra bu varsayılan kural için en az hata oranıyla istisnalar oluşturulur. Sonra her istisna için en iyi istisna kuralları üretilir. Böylece istisnaların bir ağaç gibi genişlemeleri gerçekleşir. İstisnalar varsayılan kural haricinde tahmin edilen kuralların bir kümesidir (Turna, 2011).

5.1.3. Part Algoritması

Bu algoritma, karar listesi üretmek için kullanılır. Ayır ve yönet prosedürünü kullanılır. Her iterasyonda kısmi bir C4.5 karar ağacı yapısı kurar ve en iyi yaprağı bir kural olarak ortaya konulur (Turna, 2011).

5.1.4. One-R Algoritması

One-R algoritması tek bir özelliğe dayanarak kurallar üretir. Bu algoritma ilk olarak tüm özellikleri hata oranlarına göre sıralar. Daha sonra en iyi özelliği kullanarak sınıflandırma kuralı üretir (Lavesson ve Davidsson, 2009).

5.2. Kullanılan Veritabanları

5.2.1. Pima Indians Diabetes Veritabanı

Diabet veritabanı tested_negative ve tested_positive olmak üzere 2 sınıftan oluşur. Tested_negative sınıfından 500 adet ve tested_positive sınıfından 268 adet olmak üzere toplam 768 veri bulunmaktadır. Her bir veri için 8 özellik bulunmaktadır. Özelliklerin tamamı reel değerler almaktadır. Bu özelliklerin isimleri ve hangi aralıkta değer aldıkları Tablo 5.1'de belirtilmiştir:

Tablo 5.1. Özelliklerin değer aralıkları (Diabet)

Özellik	Aldığı en küçük değer	Aldığı en yüksek değer
Preg	0	17
Plas	0	199
Pres	0	122
Skin	0	99
İnsu	0	846
Mass	0	67.1
Pedi	0.078	2.42
Age	21	81

Veritabanından örnek bir kısım ise Şekil 5.1' de gösterilmiştir.

preg Numeric	plas Numeric	pres Numeric	skin Numeric	insu Numeric	mass Numeric	pedi Numeric	age Numeric	**class** Nominal
6.0	148.0	72.0	35.0	0.0	33.6	0.627	50.0	tested_positive
8.0	125.0	96.0	0.0	0.0	0.0	0.232	54.0	tested_positive
1.0	122.0	90.0	51.0	220.0	49.7	0.325	31.0	tested_positive
1.0	163.0	72.0	0.0	0.0	39.0	1.222	33.0	tested_positive
1.0	151.0	60.0	0.0	0.0	26.1	0.179	22.0	tested_negative
0.0	125.0	96.0	0.0	0.0	22.5	0.262	21.0	tested_negative
1.0	81.0	72.0	18.0	40.0	26.6	0.283	24.0	tested_negative
2.0	85.0	65.0	0.0	0.0	39.6	0.93	27.0	tested_negative
1.0	126.0	56.0	29.0	152.0	28.7	0.801	21.0	tested_negative
1.0	96.0	122.0	0.0	0.0	22.4	0.207	27.0	tested_negative
4.0	144.0	58.0	28.0	140.0	29.5	0.287	37.0	tested_negative
3.0	83.0	58.0	31.0	18.0	34.3	0.336	25.0	tested_negative
4.0	110.0	92.0	0.0	0.0	37.6	0.191	30.0	tested_negative
0.0	95.0	85.0	25.0	36.0	37.4	0.247	24.0	tested_positive
3.0	171.0	72.0	33.0	135.0	33.3	0.199	24.0	tested_positive
8.0	155.0	62.0	26.0	495.0	34.0	0.543	46.0	tested_positive
1.0	89.0	76.0	34.0	37.0	31.2	0.192	23.0	tested_negative
4.0	76.0	62.0	0.0	0.0	34.0	0.391	25.0	tested_negative
7.0	160.0	54.0	32.0	175.0	30.5	0.588	39.0	tested_positive

Şekil 5.1. Pima indians diabetes veritabanından bir kesit

5.2.2. Ecoli Veritabanı

Ecoli veritabanı 8 sınıftan oluşmaktadır ve toplam 336 veri bulunmaktadır. Sınıflar ve o sınıfta kaç veri bulunduğu Tablo5.2'de belirtilmiştir:

Tablo 5.2. Ecoli veritabanındaki sınıflar

Sınıf	cp	im	pp	imU	om	omL	imL	imS
Veri sayısı	143	77	52	35	20	5	2	2

Ecoli veritabanında her verinin 7 özelliği bulunmaktadır. Özelliklerin tamamı reel değerler almaktadır. Bu özelliklerin isimleri ve hangi aralıkta değer aldıkları Tablo 5.3''te belirtilmiştir:

Tablo 5.3. Özelliklerin değer aralıkları (Ecoli)

Özellik	Aldığı en küçük değer	Aldığı en yüksek değer
mcg	0	0.89
gvh	0.16	1
lip	0.48	1
chg	0.5	1
aac	0	0.88
alm1	0.03	1
alm2	0	0.99

Veritabanından örnek bir kısım ise Şekil 5.2'de gösterilmiştir.

mcg Numeric	gvh Numeric	lip Numeric	chg Numeric	aac Numeric	alm1 Numeric	alm2 Numeric	class Nominal
0.61	0.47	0.48	0.5	0.0	0.8	0.32	im
0.57	0.38	0.48	0.5	0.06	0.49	0.33	imU
0.53	0.42	0.48	0.5	0.16	0.29	0.39	cp
0.72	0.86	0.48	0.5	0.17	0.55	0.21	pp
0.47	0.47	0.48	0.5	0.22	0.16	0.26	cp
0.67	0.81	0.48	0.5	0.25	0.42	0.25	pp
0.5	0.51	0.48	0.5	0.27	0.23	0.34	cp
0.54	0.47	0.48	0.5	0.28	0.33	0.42	cp
0.31	0.47	0.48	0.5	0.29	0.28	0.39	cp
0.44	0.34	0.48	0.5	0.3	0.33	0.43	cp
0.35	0.37	0.48	0.5	0.3	0.34	0.43	cp
0.69	0.67	0.48	0.5	0.3	0.39	0.24	pp
0.24	0.35	0.48	0.5	0.31	0.19	0.31	cp
0.5	0.66	0.48	0.5	0.31	0.92	0.92	im
0.66	0.74	0.48	0.5	0.31	0.38	0.43	pp
0.74	0.74	0.48	0.5	0.31	0.53	0.52	pp
0.43	0.32	0.48	0.5	0.33	0.45	0.52	cp
0.33	0.56	0.48	0.5	0.33	0.78	0.8	im
0.16	0.51	0.48	0.5	0.33	0.39	0.48	im

Şekil 5.2. Ecoli veritabanından bir kesit

5.2.3. BUPA Liver Disorders Veritabanı

BUPA veritabanıda toplam 2 sınıf bulunmaktadır. 1. sınıfta 145 adet ve 2. sınıfta 200 adet veri olmak üzere toplam 345 veri bulunmaktadır. Her bir veri için 6 özellik bulunmaktadır. Özelliklerin tamamı reel değerler almaktadır. Bu özelliklerin isimleri ve hangi aralıklarda değer aldıkları Tablo 5.4'te belirtilmiştir:

Tablo 5.4. Özelliklerin değer aralıkları (BUPA)

Özellik	Aldığı en küçük değer	Aldığı en yüksek değer
Mcv	65	103
Alkphos	23	138
Sgpt	4	155
Sgot	5	82
Gammagt	5	297
Drinks	0	20

Veritabanından örnek bir kısım ise Şekil 5.3'te gösterilmiştir.

Mcv Numeric	Alkphos Numeric	Sgpt Numeric	Sgot Numeric	Gammagt Numeric	Drinks Numeric	**Class** Nominal
96.0	55.0	48.0	39.0	42.0	4.0	2
79.0	101.0	17.0	27.0	23.0	4.0	2
90.0	134.0	14.0	20.0	14.0	4.0	2
82.0	62.0	17.0	17.0	15.0	0.5	1
89.0	76.0	14.0	21.0	24.0	4.0	2
88.0	93.0	29.0	27.0	31.0	4.0	2
92.0	73.0	24.0	21.0	48.0	4.0	2
91.0	55.0	28.0	28.0	82.0	4.0	2
83.0	45.0	19.0	21.0	13.0	4.0	2
90.0	74.0	19.0	14.0	22.0	4.0	2
92.0	66.0	21.0	16.0	33.0	5.0	1
93.0	63.0	26.0	18.0	18.0	5.0	1
86.0	78.0	47.0	39.0	107.0	5.0	2
97.0	44.0	113.0	45.0	150.0	5.0	2
86.0	77.0	25.0	19.0	18.0	0.5	1
87.0	59.0	15.0	19.0	12.0	5.0	2
86.0	44.0	21.0	11.0	15.0	5.0	2
87.0	64.0	16.0	20.0	24.0	5.0	2
92.0	57.0	21.0	23.0	22.0	5.0	2
90.0	70.0	25.0	23.0	112.0	5.0	2
99.0	59.0	17.0	19.0	11.0	5.0	2
92.0	80.0	10.0	26.0	20.0	6.0	1

Şekil 5.3. BUPA Liver Disorders veritabanından bir kesit

5.2.4. Thyroid Disease (New Thyroid) Veritabanı

New Thyroid veritabanında toplam 3 sınıf bulunmaktadır. 1. sınıfta 150 adet, 2. sınıfta 35 adet ve 3. sınıfta 30 adet olmak üzere toplam 215 veri bulunmaktadır. Her bir veri için 5 özellik bulunmaktadır ve özelliklerin tamamı reel değerler almaktadır. Bu özelliklerin isimleri ve hangi aralıklarda değer aldıkları Tablo 5.5'te belirtilmiştir:

Tablo 5.5. Özelliklerin değer aralıkları (New Thyroid)

Özellik	Aldığı en küçük değer	Aldığı en yüksek değer
T3resin	65	144
Thyroxin	0.5	25.3
Triiodothyronine	0.2	10
Thyroidstimulating	0.1	56.4
TSH_value	-0.7	56.3

Veritabanından örnek bir kısım ise Şekil 5.4'te gösterilmiştir.

T3resin Numeric	Thyroxin Numeric	Triiodothyronine Numeric	Thyroidstimulating Numeric	TSH_value Numeric	**Class** Nominal
116.0	11.9	1.8	1.9	1.5	1
116.0	11.5	1.8	1.4	5.4	1
118.0	10.6	1.8	1.4	3.0	1
109.0	9.2	1.8	1.1	4.4	1
127.0	7.7	1.8	1.9	6.4	1
104.0	6.1	1.8	0.5	0.8	1
113.0	17.2	1.8	1.0	0.0	2
94.0	20.5	1.8	1.4	-0.5	2
97.0	15.1	1.8	1.2	-0.2	2
141.0	5.6	1.8	9.2	14.4	3
121.0	4.7	1.8	11.2	53.0	3
120.0	3.4	1.8	7.5	21.5	3
112.0	8.1	1.9	3.7	2.0	1
109.0	10.4	1.9	0.4	-0.1	1
110.0	7.8	1.9	2.1	6.4	1
117.0	12.2	1.9	1.2	3.9	1
120.0	6.8	1.9	1.3	1.9	1
118.0	8.1	1.9	1.5	13.7	1
110.0	8.7	1.9	1.6	4.4	1
117.0	9.2	1.9	1.5	6.8	1
99.0	17.5	1.9	1.4	0.3	2
110.0	15.2	1.9	0.7	-0.2	2
105.0	11.1	2.0	1.0	1.0	1

Şekil 5.4. Thyroid Disease (New Thyroid) veritabanından bir kesit

5.3. Geliştirilen Uygulama

Bu tez çalışmasında, Visual C#'da POA'yı kullanarak sınıflandırma kural keşfi yapacak bir program geliştirilmiştir. POA'nın adımları şu şekilde idi:

1. Popülasyonu başlat.
2. Popülasyonu gruplara böl. En uygun üyeleri aday üye olarak al.
3. Grup içi yarışma.
    a. Gruptaki asil üyeleri aday üyelere doğru yaklaştır.
    b. Yeni aday üyeleri ata.
    c. Her grubun gücünü hesapla.
4. Gruplar arası yarışma.
    a. λ en güçlü grubu al ve onları P_m olasılıkta birleştir.
    b. Υ zayıf grubu P_d olasılıkla sil.
5. Bitim şartı sağlanmıyorsa adım 3'e dön.
6. En iyi grubun en iyi üyesini problemin çözümü olarak belirle.

Tezin bu bölümden sonraki kısımlarında, uygulamanın, algoritmanın adımlarına göre nasıl geliştirildiği ve kullanılan 4 veritabanından elde edilen sonuçlar bulunmaktadır.

5.3.1. Popülasyonun Başlatılması

Başlangıçta gelişigüzel değerler verilerek üyeler oluşturulur. Oluşturulan bu üyeler bizim başlangıç popülasyonunu oluşturacaktır. Daha sonra üyeler her grupta eşit sayıda üye olacak şekilde gelişigüzel gruplara dağıtılacaktır. Geliştirdiğimiz uygulamada başlangıç popülasyonu olarak gelişigüzel değerler alan 200 üye oluşturulmuştur.

Başlangıç popülasyonunda oluşturulan her üye aslında birer kuralı temsil eder. Başlangıçta oluşturulan her üye birer aday kuraldır. Program sonunda ise en iyi uygunluk değerine sahip olan üye problemin çözümü olacaktır.

Üyeler, kuralın sol tarafında bulunabilecek özellikleri içerir. Her özellik için ise üyeler 3 alt kısımdan oluşur. Bu kısımlar, özelliğin kuralın solunda bulunup bulunmadığını gösteren bir değer, özelliğin bulunacağı aralığın alt sınır ve üst sınır değerleridir. Bu aralık değerleri gelişigüzel bir şekilde program tarafından atanır. Şekil 5.5, bir üyenin yapısını göstermektedir.

$Ö_1$			$Ö_2$			...	$Ö_n$		
b_1	a_1	$ü_1$	b_2	a_2	$ü_2$	...	b_n	a_n	$ü_n$

Şekil 5.5. Üyelerin Yapısı

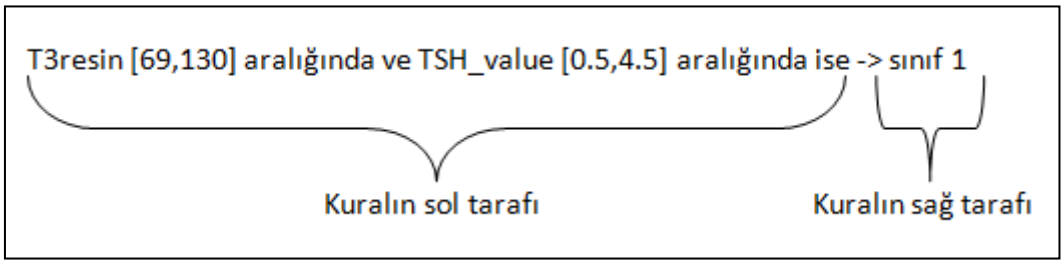
Eğer kullanılan veritabanında kaç tane özellik varsa, üyelerde okadar bölümden oluşacaktır. Bayrak (*b*), o özelliğin kuralda yer alıp almadığını, alt sınır (*a*), ilgili özelliğin değerinin bulunacağı aralığın alt sınırını ve üst sınır (*ü*), ilgili özelliğin değerinin bulunacağı aralığın üst sınırını gösterir(Alataş vd., 2006).

Örnek olarak New Thyroid veritabanından bir üyeyi ele alalım. Bu üyenin yapısını Şekil 5.6'da gösterildiği gibi olsun.

T3resin			Thyroxin			Triiodothyronine			Thyroidstimulating			TSH_value		
1	69	130	0	0.75	15.5	0	0.55	3.5	0	0.1	11	1	0.5	4.5

Şekil 5.6. New Thyroid veritabanında üye yapısı

Şekilde görüldüğü gibi New Thyroid veri tabanı 5 özellikten oluştuğu için üyede 5 bölümden oluşmaktadır. Üyenin bayrak kısmına baktığımızda, değeri 1 olan *t3resin* ve *TSH_value* özellikleri kuralımızın sol kısmında bulunacaktır. Bu üyenin yani aday kuralın sınıf 1 için olduğunu düşünürsek, kuralı Şekil 5.7'de gösterildiği gibi ifade edebiliriz.



Şekil 5.7. Aday kuralın ifade edilişi

Şekil 5.7'deki kuralı sözel olarak ifade edecek olursak; üzerinde işlem yapılan verinin t3resin özelliği 69 ile 130 arasında ve TSH_value özelliği 0.5 ile 4.5 arasında ise bu veri sınıf 1'e aittir.

5.3.2. Popülasyonun Gruplara Bölünmesi ve Aday Üyelerin Belirlenmesi

Başlangıçta 200 üye oluşturulmuştu. Bu üyeler her grupta 20 üye olacak şekilde 10 gruba eşit olarak dağıtılacaktır. Daha sonra her grupta uygunluk değeri en iyi θ sayıda üye o grubun aday üyeleri olacaktır. θ sayısını bu uygulamada grup üye sayısı / 3 olarak aldık. Yani başlangıçta gruptaki eleman sayısı 20 olduğundan 6 eleman aday üye geri kalan 14 eleman da asil üyeler olacaktır.

Bir kuralın uygunluk değeri aşağıdaki uygunluk fonksiyonu ile bulunmuştur.

$$uygunluk = c1x\frac{DP}{DP+YN}\ x\ \frac{DN}{DN+YP} -\ c2\ x\ anlaşılırlık + c3x\frac{DP}{DP+YP} \pm c4xaralıkoranı \qquad (5.1)$$

- *DP*, doğru pozitiflerdir ve kuralla aynı sınıf etiketine sahip olan, kural tarafından kapsanan örneklerin sayısıdır.
- *YP*, yanlış pozitiflerdir ve kuraldan farklı sınıf etiketine sahip olan, kural tarafından kapsanan örneklerin sayısıdır.
- *YN*, yanlış negatiflerdir ve kural tarafından kapsanmayan fakat kuralla aynı sınıf etiketine sahip örneklerin sayısıdır.
- *DN* ise doğru negatiflerdir ve kural tarafından kapsanmayan ve kuralla da aynı etikete sahip olmayan örneklerin sayısıdır.

Anlaşılırlık ise şu şekilde hesaplanır:

$$anlaşılırlık = \frac{kuralın\ solundaki\ özellik\ sayısı - 1}{veri\ tabanındaki\ tüm\ özelliklerin\ sayısı} \quad (5.2)$$

c1,*c2*,*c3* ve *c4* değerleri ise ağırlıklardır. Kullanıcı tanımlıdır ve en iyi sonucu verecek şekilde belirlenebilir (Alataş vd.,2006).

aralıkoranı ise kullanılan veritabanına göre uygunluk fonksiyonunda kullanılabilir. Bazı durumlarda kuraldaki özelliklerin alt sınır-üst sınır aralığının miktarı fazla olması durumunda daha iyi sonuç verebilir. Bazı durumlarda da bu aralık miktarının az olması durumunda daha iyi sonuç verebilir. Eğer aralık miktarı fazla olması isteniyorsa toplanma işlemi uygulanmalı, az olması isteniyorsa çıkarma işlemi uygulanmalıdır. Aralık oranı şu şekilde hesaplanır.

$$\sum_{i=1}^{i=n} \frac{ü_i - a_i}{Ö_i max - Ö_i min} \quad (5.3)$$

Burada n veritabanındaki özellik sayısıdır. $ü_i$ ve a_i işlem yapılan özelliğin kuraldaki üst ve alt sınırlarıdır. $Ö_i max$ ve $Ö_i min$ ise işlem yapılan özelliğin veritabanında aldığı maksimum ve minimum değerlerdir.

Bu çalışmada ise sadece Diabet veritabanında sınıflandırma kural keşfi yapılırken, uygunluk fonksiyonunda *aralıkoranı* kullanılmıştır. Diğer veritabanlarında uygulama fonksiyonunda *aralıkoranı* kullanılmamıştır.

Bu çalışmada, bazı veritabanları için bulunan sınıflandırma kurallarının çok az veriyi kapsadığı görülmüştür. Bu durumu engellemek için, başlangıç popülasyonu üretildikten sonra üyelerin uygunluk değerleri hesaplanırken eğer $DP<k$ ise, o üyenin değerleri $DP>k$ olana kadar gelişigüzel tekrardan atanmıştır. Bu çalışmada *k* değeri BUPA veritabanı için 7, diabet veritabanı içinse 10 olarak kullanılmıştır. Diğer 2 veritabanında ise gerek duyulmamıştır.

5.3.3. Grup İçi Yarışma

Grup içi yarışmada ise asil üyeler değerlerini aday üyelerin değerlerine doğru, 4. bölümdeki Denklem 4.16'ya göre yönelirler. Yönelmeden sonra asil üyeler, aday üyelerden daha yüksek uygunluk değerine sahip olabilir. Bu gibi durumlarda adayların yer değişikliği gerçekleşir.

Örnek olarak New Thyroid veritabanından bir grup alalım. 20 elemanlı grubun, uygunluk değerlerine göre küçükten büyüğe doğru sıralı hali Şekil 5.8'de gösterilmiştir. Grubun 1/3 en iyi uygunluk değerine sahip üyesi, yani 6 üyesi aday üye olacaktır.

üye no	T3resin			Thyroxin			Triiodothyronine			Thyroidstimulating			TSH_value		
	B1	A1	U1	B2	A2	U2	B3	A3	U3	B4	A4	U4	B5	A5	U5
0	0	101.44	119.7	1	1.17	6.48	0	2.06	7.91	0	21.41	45.26	0	38.86	42.29
7	0	98.79	120.08	1	0.5	16.61	0	0.2	7.44	0	19.9	40.3	0	35.67	41.83
11	0	100.83	102.83	1	0.5	13.74	1	2.18	6.49	0	0.1	28.58	0	29.76	36.76
22	0	90.91	118.02	1	2.44	12.65	0	4.55	5.4	0	0.1	38.36	0	27.88	34.88
32	0	96.84	111.16	1	0.69	14.61	0	1.71	10	0	10.97	52.46	0	48.61	50.09
51	1	108.96	119	1	0.5	5.5	0	2.31	7.14	0	0.1	30.33	0	29.61	34.72
63	0	81.11	104.31	1	1.67	11.62	0	3.1	6.84	0	6.77	45.21	0	25.16	32.16
73	0	96.61	114.09	1	0.5	5.5	0	3.51	9.98	0	0.1	37.72	0	29.11	36.11
96	0	112.1	114.1	1	3.5	5.5	0	2.35	7.45	0	0.1	43.05	0	28.48	35.48
97	0	102.52	124.09	1	0.5	13.22	0	0.2	8.03	0	0.1	32.39	0	29.19	36.19
127	0	70.46	112.62	1	4.92	10.32	0	1.98	5.79	0	2.49	41.75	0	18.34	25.34
150	0	69.77	118.71	1	0.5	8.88	0	3.38	6.98	0	0.1	44.07	0	37.5	42.06
151	1	80.71	110.33	1	0.5	8.12	0	0.2	7.49	0	18.26	35.77	0	37.53	39.26
192	0	96.69	117.67	1	5.1	14.85	1	0.2	8.39	0	0.1	26.5	0	30.23	37.23
1	0	136	141	1	0.996	18.98096	0	3	4.54	0	37.258	40.51214	1	24.95	40.51214
161	1	127	143	1	14.884	24.2584	0	7.1	9.71	0	19.805	44.6896	0	9.56	44.6896
26	0	125	143	0	6.7	16	0	2.1	5.576	0	25.998	44.84724	1	28.94	44.84724
39	0	108	141	0	3.972	17.62192	0	0.2	7.06	1	37.258	47.59468	0	32.93	47.59468
82	1	126	135	1	0.748	18.42544	1	2.1	9.289	0	21.494	34.06016	0	40.34	34.06016
85	0	94	118	1	3.972	10.58368	0	3	6.15	0	9.671	42.3813	0	31.79	42.3813

Asil üyeler

Aday üyeler

Şekil 5.8. Grubun ilk durumu ve seçilen aday üyeler

Asil üyelerin bayrak (*B*), alt sınır (*A*) ve üst sınır (*U*) değerleri aday üyelerin *B*, *A* ve *U* değerlerine doğru yöneleceklerdir. Bu yönelme işlemi ise uygulamamızda her döngüde tekrarlanacaktır.

Şekil 5.9'da ise aynı grubun bir döngü sonraki durumu, uygunluk değerlerine göre küçükten büyüğe doğru sıralı bir şekilde gösterilmiştir.

üye no	T3resin			Thyroxin			Triiodothyronine			Thyroidstimulating			TSH_value		
	B1	A1	U1	B2	A2	U2	B3	A3	U3	B4	A4	U4	B5	A5	U5
1	0	73.58	119.47	1	3.32	11.72	0	1.68	7.78	0	17.44	38	0	28.69	34.92
161	0	75.74	88.87	1	1.81	12.26	0	0.2	5.08	0	6.24	31.66	0	34.4	34.46
26	0	69.8	118.8	1	2.98	13.48	0	2.2	6.74	0	0.1	36.06	0	26.07	27.28
39	0	85.65	104.55	1	2.49	7.29	0	2.51	6.52	0	19.61	33.68	0	21.14	38.06
73	0	85.14	115.32	1	5	12.5	0	1.44	5.94	0	3.71	37.87	0	26.91	33.32
51	0	98.18	116.09	1	3.09	16.47	0	2.16	6.5	0	4.38	36.98	0	27.75	34.67
96	0	89.32	115.68	1	2.99	16.71	0	2.16	6.12	0	6.44	34.85	0	27.38	34.46
0	0	90.41	117.34	1	3.29	17.72	0	2.28	6.09	0	9.15	41.38	0	17.2	35.97
82	0	78.02	108.71	1	4.61	14.35	0	2.23	5.63	0	0.1	38.65	0	25.02	34.65
150	0	84.55	115.1	1	4.87	12.03	0	1.51	6.55	0	2.26	34.74	0	19.83	31.32
11	0	79.4	122.3	1	3.2	11.85	0	2.19	6.77	0	5.16	44.3	0	25.85	32.92
7	0	79.29	116.5	1	3.27	8.12	0	2.07	6.93	0	0.1	36.19	0	21.47	31.79
151	0	88.85	118.48	1	3.44	13.8	0	3.33	6.99	0	0.1	37.4	0	19.94	36.8
32	0	84.1	114.19	1	5.13	12.9	0	2.14	6.21	0	4.49	32.47	0	13.39	32.9
97	0	102.52	124.09	1	0.5	13.22	0	0.2	8.03	0	0.1	32.39	0	29.19	36.19
63	0	81.11	104.31	1	1.67	11.62	0	3.1	6.84	0	6.77	45.21	0	25.16	32.16
192	0	96.69	117.67	1	5.1	14.85	1	0.2	8.39	0	0.1	26.5	0	30.23	37.23
22	0	90.91	118.02	1	2.44	12.65	0	4.55	5.4	0	0.1	38.36	0	27.88	34.88
85	0	94	118	1	3.972	10.58368	0	3	6.15	0	9.671	42.3813	0	31.79	42.3813
127	0	70.46	112.62	1	4.92	10.32	0	1.98	5.79	0	2.49	41.75	0	18.34	25.34

Asil üyeler

Aday üyeler

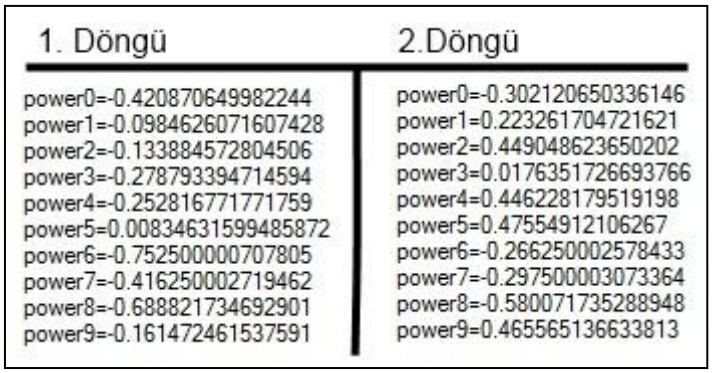
Şekil 5.9. Grubun bir döngü sonraki durumu ve seçilen aday üyeler

Şekil 5.8 ve Şekil 5.9'a bakıldığında, yönelme sonrası asil üyelerin *B,A* ve *U* değerlerinin değiştiği görülebilir. Ayrıca 5 tane asil üyenin uygunluk değerleri, 85 numaralı aday üye dışındaki diğer aday üyelerden daha iyi bir değere ulaştığı için adaylarda yer değişikliği yapıldığı da rahatlıkla görülebilmektedir.

Yönelme sonrası her grubun gücü hesaplanır ve daha sonra algoritmada gruplar arası yarışma adımına geçilir. Her grubun gücü 4. bölümde de bahsedilen aşağıdaki formül ile hesaplanır.

$$power^i = \frac{rn1\ x\ ort(Q^i) + rn2\ x\ ort(R^i)}{rn1+rn2}\ ; rn1 > rn2 \qquad (5.4)$$

Denklem 5.4'te $ort(Q^i)$ ve $ort(R^i)$ sırasıyla gruptaki aday üyelerin ve asil üyelerin değerlerinin ortalamasıdır. *rn1* ve *rn2* ise gelişigüzel değerlerdir. $power^i$ ise *i.* grubun gücünü ifade eder. Bu uygulamada *rn1* ve *rn2* değerleri sırası ile 9 ve 3 alınmıştır. Uygulamada her döngü sonrası grupların gücü tekrar hesaplanır. Şekil 5.10'da 1. ve 2. döngü sonrasındaki grupların gücü görülmektedir.



Şekil 5.10. Grupların güçleri

Şekil incelendiğinde grup güçlerinin bir döngü sonrasında arttığı görülmektedir. Buradan her gruptaki üyelerin uygunluk değerlerinin arttığı yani kuralların daha iyiye doğru gittiği sonucu çıkarılabilir.

5.3.4. Gruplar Arası Yarışma

Gruplar arası yarışmada ise λ sayıda en güçlü grup alınır ve P_m olasılıkta birleştirilir. Υ sayıda zayıf grup ise P_d olasılıkla silinir. Bu uygulamada birleşme durumu, en güçlü 2 grup yüzde 3 olasılıkta birleşecek şekilde ayarlanmıştır. Silme durumu ise en güçsüz grup yüzde 1 olasılıkta silinecek şekilde ayarlanmıştır.

5.3.5. Bitim Şartı

Bu uygulamada bitim şartı 100 döngü olarak alınmıştır. 100 döngü sonunda en yüksek güce sahip grubun en yüksek uygunluk değerine sahip üyesi bizim çözümümüz yani sınıflandırma kuralımız olacaktır.

5.3.6. Uygulama Sonuçları

Bu bölümde her veritabanı için kullanılan parametre değerleri, yapılan programdan elde edilen sonuçlar ve WEKA programında elde edilen sonuçlar tablolar halinde gösterilecektir. Sınıflandırma kuralları keşfedilirken kullanılan veri tabanındaki veriler, eğitim verisi ve test verisi olarak ayrılmadan verilerin tamamı eğitim verisi olarak kullanılmıştır. Yine bulunan kurallar verilerin tamamı üzerinde test edilmiştir.

5.3.6.1. Pima Indians Diabetes Veritabanı Sonuçları

Diabet veri tabanında kullanılan parametre değerleri Tablo 5.6'da gösterilmiştir.

Tablo 5.6. Diabet veritabanında kullanılan parametre değerleri

Parametre	c1	c2	c3	c4
Değer	0.28	0.20	0.40	0.12

POA tabanlı geliştirilen uygulamadan elde edilen sonuçlar ise Tablo 5.7'de gösterilmiştir.

Tablo 5.7. POA' da elde edilen sonuçlar (Pima Indians Diabetes)

	Toplam bulunan kural sayısı	Toplam doğru veri sayısı / tüm verilerin sayısı	Doğruluk yüzdesi
1.Çalıştırmada	13	611/768	%79,55
2.Çalıştırmada	13	609/768	%79,29
3.Çalıştırmada	6	609/768	%79,29
4.Çalıştırmada	7	600/768	%78.12

Diabet veri tabanı için programın 4 defa çalıştırılmasından elde edilen kuralların, ortalama %79.06 oranında verileri doğru sınıflandırdığı görülmüştür. Örnek olarak 3. çalıştırma ele alındığında, elde edilen 6 kural aşağıda gösterildiği gibidir.

1. (40≤plas≤127.99) ise sınıf=tested_negative.
2. (55.81≤plas≤145.51) ve (21.85≤mass≤28.27) ise sınıf=tested_negative.
3. (0≤preg≤1.26) ve (63.66≤plas≤152.33) ve (69.98≤pres≤86.5) ve (18.08≤mass≤45.09) ise sınıf=tested_negative.
4. (38.5≤pres≤110) ve (18.05≤mass≤29.97) ise sınıf=tested_negative.
5. (40≤plas≤148.03) ve (72.06≤pres≤101.26) ve (18.98≤mass≤53.1) ve (0≤pedi≤0.39) ise sınıf=tested_negative.
6. Yukarıdaki 5 kural dışında ise sınıf=tested_positive.

WEKA programında elde edilen sonuçlar ise Tablo 5.8'de gösterilmiştir.

Tablo 5.8. WEKA programında elde edilen sonuçlar (Pima Indians Diabetes)

Kullanılan algoritma	Toplam bulunan kural sayısı	Toplam doğru veri sayısı / tüm verilerin sayısı	Doğruluk yüzdesi
Jrip	4	609/768	%79,29
Ridor	4	605/768	%78,77
Part	13	624/768	%81,25
One-R	10	586/768	%76,30

WEKA'da ise örnek olarak Jrip algoritması seçildiğinde, bulduğu 4 kural aşağıda gösterildiği gibidir.

1. (plas ≥ 132) ve (mass ≥ 30) ise sınıf=tested_positive.
2. (age ≥ 29) ve (insu ≥ 125) ve (preg ≤ 3) ise sınıf=tested_positive.
3. (age ≥ 31) ve (pedi ≥ 0.529) ve (preg ≥ 8) ve (mass ≥ 25.9) ise sınıf=tested_positive.
4. Yukarıdaki 3 kural dışında ise sınıf=tested_negative.

5.3.6.2. Ecoli Veritabanı Sonuçları

Ecoli veritabanında kullanılan parametre değerleri Tablo 5.9'da gösterilmiştir.

Tablo 5.9. Ecoli veritabanında kullanılan parametre değerleri

Parametre	c1	c2	c3
Değer	0.20	0.20	0.60

POA tabanlı geliştirilen uygulamadan elde edilen sonuçlar ise Tablo 5.10'da gösterilmiştir.

Tablo 5.10. POA' da elde edilen sonuçlar (Ecoli)

	Toplam bulunan kural sayısı	Toplam doğru veri sayısı / tüm verilerin sayısı	Doğruluk yüzdesi
1.Çalıştırmada	12	285/336	%84,82
2.Çalıştırmada	22	290/336	%86,70
3.Çalıştırmada	12	274/336	%81,54
4.Çalıştırmada	18	281/336	%83,63

Ecoli veri tabanı için programın 4 defa çalıştırılmasından elde edilen kuralların, ortalama %84.17 oranında verileri doğru sınıflandırdığı görülmüştür. Örnek olarak 1. çalıştırma ele alındığında, elde edilen 12 kural aşağıda gösterildiği gibidir.

1. (0.14≤mcg≤0.55) ve (0.22≤alm2≤0.61) ise sınıf=cp.
2. (0.16≤gvh≤0.42) ve (0.1≤aac≤0.88) ve (0≤alm2≤0.6) ise sınıf=cp.
3. (0.06≤ mcg≤0.61) ve (0.18≤gvh≤0.87) ve (0.05≤aac≤0.88) ve (0.6≤alm2≤0.97) ise sınıf=im.
4. (0.57≤gvh≤0.7) ve (0.18≤aac≤0.88) ve (0.55≤alm1≤0.89) ve (0.22≤alm2≤0.93) ise sınıf=im.
5. (0.28≤mcg≤0.71) ve (0.68≤aac≤0.88) ve (0.63≤alm1≤0.84) ve (0≤alm2≤0.78) ise sınıf=im.
6. (0.63≤mcg≤0.66) ve (0.52≤gvh≤0.81) ve (0.39≤aac≤0.88) ve (0.57≤alm1≤0.8) ve (0.22≤alm2≤0.97) ise sınıf=im.
7. (0.36≤gvh≤0.56) ve (0.3≤aac≤0.88) ve (0.91≤alm2≤0.99) ise sınıf=im.
8. (0.25≤aac≤0.88) ve (0.57≤alm≤0.81) ve (0.57≤alm2≤0.89) ise sınıf=imU.
9. (0.46≤gvh≤0.48) ve (0.1≤aac≤0.68) ve (0.17≤alm2≤0.99) ise sınıf=om.
10. (0.57≤gvh≤0.89) ve (0.23≤aac≤0.88) ve (0.29≤alm1≤0.63) ve (0.16≤alm2≤0.94) ise sınıf=pp.
11. (0.48≤mcg≤0.78) ve (lip=0.48) ve (0.48≤alm1≤0.69) ve (0.35≤alm2≤0.95) ise sınıf=pp.
12. (0.31≤mcg≤0.72) ve (0.37≤gvh≤1) ve (0.21≤aac≤0.88) ve (0.46≤alm1≤0.68) ise sınıf=omL.

WEKA programında elde edilen sonuçlar ise Tablo 5.11'de gösterilmiştir.

Tablo 5.11. WEKA programında elde edilen sonuçlar (Ecoli)

Kullanılan algoritma	Toplam bulunan kural sayısı	Toplam doğru veri sayısı / tüm verilerin sayısı	Doğruluk yüzdesi
Jrip	10	305/336	%90,77
Ridor	46	297/336	%88,39
Part	13	308/336	%91,66
One-R	7	232/336	%69,04

WEKA'da ise örnek olarak One-R algoritması seçildiğinde, bulduğu 7 kural aşağıda gösterildiği gibidir.

1. alm1< 0.395 ise sınıf = cp.
2. alm1< 0.425 ise sınıf = pp.
3. alm1< 0.46499999999999997 ise sınıf = cp.
4. alm1< 0.575 ise sınıf = pp.
5. alm1< 0.715 ise sınıf = im.
6. alm1< 0.745 ise sınıf = imU.
7. alm1≥0.745 ise sınıf = im.

5.3.6.3. BUPA Liver Disorders Veritabanı Sonuçları

BUPA veritabanında kullanılan parametre değerleri Tablo 5.12'de gösterilmiştir.

Tablo 5.12. BUPA veritabanında kullanılan parametre değerleri

Parametre	c1	c2	c3
Değer	0.20	0.20	0.60

POA tabanlı geliştirilen uygulamadan elde edilen sonuçlar ise Tablo 5.13'te gösterilmiştir.

Tablo 5.13. POA' da elde edilen sonuçlar (BUPA)

	Toplam bulunan kural sayısı	Toplam doğru veri sayısı / tüm verilerin sayısı	Doğruluk yüzdesi
1.Çalıştırmada	16	276/345	%80,00
2.Çalıştırmada	14	276/345	%80,00
3.Çalıştırmada	15	273/345	%79,13
4.Çalıştırmada	15	270/345	%78,26

BUPA veri tabanı için programın 4 defa çalıştırılmasından elde edilen kuralların, ortalama %79.34 oranında verileri doğru sınıflandırdığı görülmüştür. Örnek olarak 2. çalıştırma ele alındığında, elde edilen 14 kural aşağıda gösterildiği gibidir.

1. (45.6≤sgot≤56.25) ise sınıf=2.
2. (70.4≤mcv≤89.47) ve (41.96≤gammagt≤62.2) ise sınıf=2.
3. (3.54≤drinks≤5.54) ise sınıf=2.

4. (82.44≤mcv≤85.67) ve (11.62≤sgot≤43.18) ise sınıf=2.
5. (24.41≤sgot≤47.26) ve (36.94≤gammagt≤55.1) ise sınıf=2.
6. (90.84≤mcv≤96.85) ve (1.91≤drinks≤2) ise sınıf=2.
7. (4≤sgpt≤71.15) ve (9.99≤drinks≤12.18) ise sınıf=2.
8. (79.99≤mcv≤90.52) ve (5.41≤drinks≤13.86) ise sınıf=2.
9. (83.73≤mcv≤95.4) ve (26.12≤alkphos≤51.63) ve (21.81≤sgot≤48.6) ise sınıf=2.
10. (55.46≤alkphos≤57.08) ise sınıf=2.
11. (11.79≤sgpt≤16.64) ise sınıf=2.
12. (51.98≤alkphos≤94.05) ve (35.52≤gammagt≤40.69) ise sınıf=2.
13. (46.18≤alkphos≤84.27) ve (64.14≤gammagt≤87.64) ise sınıf=2.
14. Yukarıdaki 13 kural dışında ise sınıf=1.

WEKA programında elde edilen sonuçlar ise Tablo 5.14'te gösterilmiştir.

Tablo 5.14. WEKA programında elde edilen sonuçlar (BUPA)

Kullanılan algoritma	Toplam bulunan kural sayısı	Toplam doğru veri sayısı / tüm verilerin sayısı	Doğruluk yüzdesi
Jrip	5	270/345	%78,26
Ridor	3	246/345	%71,30
Part	15	297/345	%86,08
One-R	14	235/345	%68,11

WEKA' da ise örnek olarak Ridor algoritması seçildiğinde, bulduğu 3 kural aşağıda gösterildiği gibidir.

1. (gammagt ≤ 35.5) ve (sgpt > 21.5) ve (sgot ≤ 20.5) ise sınıf=1.
2. (drinks > 5.5) ve (sgpt > 36.5) ise sınıf=1.
3. (alkphos > 61.5) ve (gammagt ≤ 20.5) ve (sgpt > 19.5) ve (sgpt > 24.5) ise sınıf=1.

5.3.6.4. Thyroid Disease (New Thyroid) Veritabanı Sonuçları

Thyroid veritabanında kullanılan parametre değerleri Tablo 5.15'de gösterilmiştir.

Tablo 5.15. Thyroid veritabanında kullanılan parametre değerleri

Parametre	c1	c2	c3
Değer	0.20	0.20	0.60

POA tabanlı geliştirilen uygulamadan elde edilen sonuçlar ise Tablo 5.16'da gösterilmiştir.

Tablo 5.16. POA' da elde edilen sonuçlar (Thyroid)

	Toplam bulunan kural sayısı	Toplam doğru veri sayısı / tüm verilerin sayısı	Doğruluk yüzdesi
1.Çalıştırmada	7	211/215	%98,13
2.Çalıştırmada	5	205/215	%95,34
3.Çalıştırmada	8	201/215	%93,48
4.Çalıştırmada	5	206/215	%95,81

Thyroid veri tabanı için programın 4 defa çalıştırılmasından elde edilen kuralların, ortalama %95,69 oranında verileri doğru sınıflandırdığı görülmüştür. Örnek olarak 1. çalıştırma ele alındığında, elde edilen 7 kural aşağıda gösterildiği gibidir.

1. (6.56≤Thyroxin≤11.9) ise sınıf=1.
2. (106.33≤T3resin≤144) ve (11.14≤Thyroxin≤14.28) ise sınıf=1.
3. (97.51≤T3resin≤101.78) ve (0.6≤TSH_value≤36.96) ise sınıf=1.
4. (1.39≤TSH_value≤54.75) ve (6.81≤Thyroxin≤16.1) ise sınıf=1.
5. (103.21≤T3resin≤112.99) ve (3.77≤Thyroxin≤11.44) ise sınıf=1.
6. (8.73≤Thyroxin≤25.3) ise sınıf=2.
7. Yukarıdaki 6 kural dışında ise=sınıf=3.

WEKA programında elde edilen sonuçlar ise Tablo 5.17'de gösterilmiştir.

Tablo 5.17. WEKA programında elde edilen sonuçlar (Thyroid)

Kullanılan algoritma	Toplam bulunan kural sayısı	Toplam doğru veri sayısı / tüm verilerin sayısı	Doğruluk yüzdesi
Jrip	4	209/215	%97,20
Ridor	7	206/215	%95,81
Part	4	213/215	%99,06
One-R	3	198/215	%92,09

WEKA' da ise örnek olarak Part algoritması seçildiğinde, bulduğu 4 kural aşağıda gösterildiği gibidir.

1. (Thyroidstimulating ≤ 3.7) ve (Triiodothyronine ≤ 2.9) ve (Thyroxin ≤ 14.3) ve (Thyroxin > 5.3) ve (T3resin > 95) ise sınıf=1.
2. TSH_value <= 0.8 ise sınıf=2.
3. Thyroxin ≤ 7.1 ve TSH_value > 1.8 ise sınıf=3.
4. Yukarıdaki 3 kural dışında ise sınıf=1.

6. SONUÇ

Bu tez çalışmasında Visual C# programında, POA ile sınıflandırma kural keşfi yapacak program geliştirilmiştir. UCI veri ambarından alınan 4 farklı veri tabanı bu programda uygulanmış ve sınıflandırma kuralları elde edilmiştir.

WEKA programında elde edilen sonuçlar ile bizim elde ettiğimiz sonuçları karşılaştırdığımızda:

Diabet veritabanı için elde edilen kurallar, ortalama %79.06 oranında verileri doğru sınıflandırmıştır. WEKA' da ise Jrip %79.29, Ridor %78.77, Part %81.25 ve One-r %76.30 doğrulukta verileri sınıflandırmıştır. Sonuçlar karşılaştırıldığında, POA, Jrip ve Ridor algoritmalarıyla yaklaşık aynı oranda doğru sonuç bulmuştur. One-r algoritmasından daha iyi, Part algoritmasından ise daha kötü sonuçlar elde edilmiştir.

Ecoli veritabanı için elde edilen kurallar, ortalama %84.17 oranında verileri doğru sınıflandırmıştır. WEKA' da ise Jrip %90.77, Ridor %88.39, Part %91.66 ve One-r %69.04 doğrulukta verileri sınıflandırmıştır. Sonuçlar karşılaştırıldığında sadece One-r algoritmasından daha iyi değerler elde edilmiştir.

BUPA veritabanı için elde edilen kurallar, ortalama %79.34 oranında verileri doğru sınıflandırmıştır. WEKA' da ise Jrip %78.26, Ridor %71.30, Part %86.08 ve One-r %68.11 doğrulukta verileri sınıflandırmıştır. Sonuçlar karşılaştırıldığında Part algoritması dışındaki algoritmalardan daha iyi değerler elde edilmiştir.

Thyroid veritabanı için elde edilen kurallar, ortalama %95.69 oranında verileri doğru sınıflandırmıştır. WEKA' da ise Jrip %97.20, Ridor %95.81, Part %99.06 ve One-r %92.09 doğrulukta verileri sınıflandırmıştır. Bu veritabanında bütün algoritmalarında yüksek değerde sonuçlar elde edildiği görülmüştür. Fakat sonuçları karşılaştırırsak Ridor ve Jrip algoritmalarıyla yaklaşık aynı oranda doğru sonuç elde edilmiştir. One-r algoritmasından daha iyi, Part algoritmasından ise daha kötü sonuçlar elde edilmiştir.

Kullanılan veri tabanları için diğer algoritmalar da (Jrip, Ridor, Part, One-r) göz önünde bulundurulduğunda, bu tez çalışması göstermektedir ki POA, sınıflandırma kural

keşfinde etkili bir yöntemdir. Önerilen yöntem üzerinde fazla bir optimizasyon yapılmadığı halde doğrulukları yüksek kurallar elde edilmiştir.

KAYNAKLAR

Abdechiri, M., Faez, K. and Bahrami, H., 2010. Adaptive Imperialist Competitive Algorithm, 9th IEEE International Conference on Cognitive Informatics (ICCI), 940.

Abdechiri, M., Faez, K. and Bahrami, H., 2010. Neural Network Learning Based on Chaotic Imperialist Competitive Algorithm, 2nd International Workshop on Intelligent Systems and Applications (ISA), pp.1-5.

Akpınar, H., 2000. Veri Tabanlarında Bilgi Keşfi ve Veri Madenciliği, İ.Ü. İşletme Fakültesi Dergisi, Cilt 29, Sayı 1/Nisan, s. 1-22.

Alataş, B., 2011. ACROA: Artificial Chemical Reaction Optimization Algorithm for global optimization, Expert Syst. Appl. 38(10): 13170-13180

Alataş, B., 2007. Kaotik haritalı parçacık sürü optimizasyonu algoritmaları geliştirme, *Doktora Tezi*, Fırat Üniversitesi Fen Bilimleri Enstitüsü.

Alataş, B., Karcı, A. ve Akın, E., 2006. Sınıflandırma Kurallarının Parçacık Sürü Optimizasyon Algoritmasıyla Keşfi, 62-66, ASYU-INISTA 2006, İstanbul.

Atashpaz-Gargari, E. and Lucas, C., 2007. Imperialist Competitive Algorithm: An Algorithm for Optimization Inspired by Imperialistic Competition, IEEE Congress on Evolutionary Computation, CEC 2007, 4661-4667.

Birbil, I. and Fang, S., 2003. An Electromagnetism-like Mechanism for Global Optimization, Journal of Global Optimization, 25, 263-282.

Borji, A., 2007. A New Global Optimization Algorithm Inspired by Parliamentart Political Competitions, Lecture Notes in Computer Science, 4827/2007, 61-71.

Borji, A. and Hamidi, M., 2009. A New Approach to Global Optimization Motivated by Parliamentary Political Competitions. Int. Journal of Innovative Computing, Information and Control, Vol. 5, No. 6, 1643-1653.

Cura, T., 2008. Modern Sezgisel Teknikler ve Uygulamaları, Papatya Yayıncılık Eğitim.

Delice, Y., 2008. Parçacık Sürü Optimizasyonu ile Yapay Sinir Ağlarından Sınıflandırma Kuralı Çıkarımı, *Yüksek Lisans Tezi*, Erciyes Üniversitesi Sosyal Bilimler Enstitüsü.

Demirel, B., 2010. Veri Madenciliğinde Chaid Algoritmasının Sosyal Güvenlik Kurumu Veri Tabanına Uygulanması, *Yüksek Lisans Tezi*, Gazi Üniversitesi Fen Bilimleri Enstitüsü.

Dorigo M., Maniezzo, V. and Colorni, A., 1991. The Ant System: An Autocatalytic Optimizing Process. Tech. Rep. No. 91- 016, Dipartimento di Elettronica, Politecnico di Milano, Italy.

Geem, Z.W, Kim, J.H. and Loganathan, G.V., 2001. A new heuristic optimization algorithm: harmony search, Simul.-T. Soc. Mod. Sim., 76(2), 60-68.

Ghalehpardaz, S.L. and Shafiee , M., 2011. Speed Control of DC Motor Using Imperialist Competitive Algorithm Based on PI-Like FLC, Third International Conference on Computational Intelligence, Modelling and Simulation, pp. 28-33.

Glover, F., 1989. Tabu Search-Part I, ORSA Journal on Computing, 1, 3, 190-206,

Han J. and Kamber M., 2001. Data mining: Concepts and techniques, Morgan Kaufmann Publishers, San Francisco.

Holland, J. H., 1975. Adaption in Natural and Artificial Systems, University of Michigan Pres, Ann Arbor, MI.

Karaboğa, D., 2004. Yapay Zeka Optimizasyon Algoritmaları, Nobel Yayın Dağıtım.

Kennedy, J. and Eberhart, R. C., 1995. Particle swarm optimization, Proc. IEEE int'l conf. on neural networks Vol. IV, pp. 1942-1948. IEEE service center, Piscataway, NJ.

Kirkpatrick, S., Gerlatt, C.D., and Vecchi, M.P., 1983. Optimization by simulated annealing, Science, Vol. 220, 671-680.

Khademolghorani, F., 2011. An effective algorithm for mining association rules based on imperialist competitive algorithm, Sixth International Conference on Digital Information Management pp. 6-11.

Lavesson, N. and Davidsson, P., 2009. AMORI:A Metric-Base One Rule Inducer, SIAM International Conference on Data Mining, 930-941.

Movahed, M.A. and Yazdani, A.M., 2011. Application of Imperialist Competitive Algorithm in Online PI Controller, Second International Conference on Intelligent Systems, Modelling and Simulation, pp. 83-87.

Özkan, Y., 2008. Veri Madenciliği Yöntemleri, Papatya Yayıncılık Eğitim.

Rao, R.V., Savsani, V.J. and Vakharia, D.P., 2012. Teaching–Learning-Based Optimization: An optimization method for continuous non-linear large scale problems, Information Sciences, vol. 183, no. 1, pp. 1-15,

Razzaghpour, M. and Rusu, A., 2011. Analog circuit optimization via a modified Imperialist Competitive Algorithm, IEEE International Symposium on Circuits and Systems, pp. 2273-2276.

Sayadnavard, M.H., Haghighat, A.T. and Abdechiri, M., 2010. Wireless sensor network localization using Imperialist Competitive Algorithm, 3rd IEEE International Conference on Computer Science and Information Technology, vol. 9, pp.818-822.

Storn, R. and Price, K., (1995), Differential Evolution: A Simple and Efficient Adaptive Scheme for Global Optimization overContinuous Spaces, Technical Report TR-95-012, International Computer Science Institute, Berkeley.

Tamimi, A., Sadjadian, H. and Omranpour, H., 2010. Mobile Robot Global Localization using Imperialist Competitive Algorithm, 3rd International Conference on Advanced Computer Theory and Engineering, vol.5, no. V5-524-V5-529, pp. 20-22.

Tiryaki, S., 2006. Lojistik Alanında Bir Veri Madenciliği Uygulaması, *Yüksek Lisans Tezi*, İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü.

Turna, F., 2011. Veri Madenciliği Teknikleriyle Tramvay Arıza Kayıtlarından Kural Çıkarımı, *Yüksek Lisans Tezi*, Erciyes Üniversitesi Fen Bilimleri Enstitüsü.

URL-1, http://archive.ics.uci.edu/ml/datasets/Diabetes Diabet veritabanı. 15 Haziran 2012.

URL-2, http://archive.ics.uci.edu/ml/datasets/Ecoli Ecoli veritabanı. 3 Ağustos 2012.

URL-3, http://archive.ics.uci.edu/ml/datasets/Liver+Disorders BUPA Liver Disorders veritabanı, 20 Ağustos 2012.

URL-4, http://sci2s.ugr.es/keel/dataset.php?cod=66 New Thyroid veritabanı, 5 Eylül 2012.

URL-5, http://www.cs.waikato.ac.nz/~ml/weka/ WEKA verimadenciliği programı, 15 Haziran 2012.

ÖZGEÇMİŞ

Soner Kızıloluk, 01.11.1987'de Sivas'ta doğdu. İlk, orta ve lise eğitimini Sivas'ta tamamladı. 2005 yılında Sivas Lisesinden mezun oldu. 2005 yılında başladığı Mersin Üniversitesi Bilgisayar Mühendisliği bölümünü 2009 yılında bitirdi. 2009 yılında, Tunceli Üniversitesi Mühendislik Fakültesi Bilgisayar Mühendisliği bölümünde araştırma görevlisi olarak işe başladı. Halen aynı görevi sürdürmektedir.