T. C.
ERCİYES ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

KARINCA KOLONİ OPTİMİZASYONU İLE YAPAY SİNİR AĞLARINDAN KURAL ÇIKARIMI

Tezi Hazırlayan
Sinem KULLUK

Tezi Yöneten
Prof. Dr. Hüseyin YAPICI
Yrd. Doç. Dr. Lale ÖZBAKIR

Makine Mühendisliği Anabilim Dalı
Doktora Tezi

Nisan 2009
KAYSERİ

T. C.
ERCİYES ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

KARINCA KOLONİ OPTİMİZASYONU İLE YAPAY SİNİR AĞLARINDAN KURAL ÇIKARIMI

Tezi Hazırlayan
Sinem KULLUK

Tezi Yöneten
Prof. Dr. Hüseyin YAPICI
Yrd. Doç. Dr. Lale ÖZBAKIR

Makine Mühendisliği Anabilim Dalı
Doktora Tezi

Bu çalışma Erciyes Üniversitesi Bilimsel Araştırma Projeleri Birimi tarafından FBT-07-61 kodlu proje ile desteklenmiştir.

Nisan 2009
KAYSERİ

Prof. Dr. Hüseyin YAPICI ve Yrd. Doç. Dr. Lale ÖZBAKIR danışmanlığında **Sinem KULLUK** tarafından hazırlanan "**Karınca Koloni Optimizasyonu ile Yapay Sinir Ağlarından Kural Çıkarımı**" adlı bu çalışma, jürimiz tarafından Erciyes Üniversitesi Fen Bilimleri Enstitüsü Makine Mühendisliği Anabilim Dalında **Doktora** tezi olarak kabul edilmiştir.

17/04/2009

JÜRİ:

Başkan:Prof. Dr. Rızvan EROL

Üye : Prof. Dr. Hüseyin YAPICI

Üye : Prof. Dr. Adil BAYKASOĞLU

Üye : Doç. Dr. S. Orhan AKANSU

Üye : Doç. Dr. Adem KALINLI

ONAY:

Bu tezin kabulü, Enstitü Yönetim Kurulunun 21.04.2009 tarih ve 2009/14-13 sayılı kararı ile onaylanmıştır.

21/04/2009

Prof. Dr. Nusret AYYILDIZ

Enstitü Müdürü

TEŞEKKÜR

Çalışmalarım boyunca, tez konumun belirlenmesi ve yürütülmesinde değerli görüş ve katkılarıyla beni yönlendiren, her konuda desteğini esirgemeyen, kıymetli tecrübelerinden faydalandığım hocalarım başta tez danışmanlarım Sayın Prof. Dr. Hüseyin YAPICI'ya, Sayın Yrd. Doç. Dr. Lale ÖZBAKIR'a ve Sayın Prof. Dr. Adil BAYKASOĞLU'na teşekkürü bir borç bilirim. Ayrıca EÜBAP-FBT-07-61 kodlu proje ile çalışmalarıma maddi destek sağlayan Erciyes Üniversitesi, Bilimsel Araştırma Projeleri Birimine teşekkür ederim.

Doktora çalışmamın her aşamasında beni teşvik eden, anlayış ve desteği ile her zaman yanımda olan eşim Mustafa KULLUK ve bugüne gelmemde büyük emeği geçen CİNGÖZ ve KULLUK ailelerine saygı ve teşekkürlerimi sunarım.

Her an gülen yüzüyle tüm yorgunluğumu alan, yanımda olarak bana çalışma azmi veren biricik kızımız Elif Berru'ya ve çalışmalarım sırasında bana destek olan tüm arkadaşlarıma teşekkür ederim.

KARINCA KOLONİ OPTİMİZASYONU İLE YAPAY SİNİR AĞLARINDAN KURAL ÇIKARIMI

Sinem KULLUK

Erciyes Üniversitesi, Fen Bilimleri Enstitüsü
Doktora Tezi, Nisan, 2009
Tez Danışmanları: Prof. Dr. Hüseyin YAPICI
Yrd. Doç. Dr. Lale ÖZBAKIR

ÖZET

Veriden sınıflandırma kuralları çıkarımı, veri madenciliği (VM) uygulamalarının ilgi çeken ve önemli bir işidir. Günümüze kadar literatürde sınıflandırma kural çıkarımına, etkin algoritmalar geliştirmek için birçok çalışma yapılmıştır. Konu, veri madenciliği alanındaki merkezi rolünden dolayı halen araştırmacıların ilgisini çekmektedir.

Yapay sinir ağları (YSA), veri madenciliğinin sınıflandırma işinde en çok kullanılan tekniklerden biridir. YSA'lar yüksek sınıflandırma doğrulukları elde edebilmelerine rağmen, tanımlama yeteneklerinin olmaması önemli bir eksiklikleridir. YSA'lar kara kutulardır ve bir problemi nasıl öğrendiklerini ve çözdüklerini anlamak çok zordur. Yapay sinir ağlarının sınıflandırma problemlerinde kullanılmasındaki temel neden, bu modellerden anlaşılır bilgiyi çıkarmaktır.

Bu amaçla tez çalışmasında sınıflandırma problemleri için, eğitilmiş yapay sinir ağlarından bilgi kazanımına yönelik bir algoritma geliştirilmiştir. Geliştirilen algoritma, YSA yapısında bağlantı ağırlıkları formunda bulunan gizli bilgiyi keşfetmek için eğitilmiş yapay sinir ağları üzerinde çalışmaktadır. Önerilen algoritma, temelde Tur Atan Karınca Koloni Optimizasyon Algoritması (TAKKO) olarak bilinen bir meta-sezgisele dayanmaktadır ve iki-adımlı hiyerarşik bir yapıya sahiptir. İlk adımda çok katmanlı algılayıcı tipi sinir ağı eğitilmekte ve ağırlıkları çıkarılmaktadır. Ağırlıklar elde edildikten sonra, ikinci adımda TAKKO algoritması sınıflandırma kurallarının üretimi için kullanılmaktadır.

Önerilen algoritma deneysel olarak on iki ikili ve çok-sınıflı referans veri kümesinde analiz edilip, değerlendirilmiştir. Bu deneysel çalışmalar ve diğer klasik ve modern kural çıkarım algoritmalar ile karşılaştırmalar, geliştirilen yaklaşımın doğru ve özlü sınıflandırma kuralları keşfetmekte büyük potansiyeli olduğunu göstermektedir.

Anahtar Kelimeler: Veri Madenciliği, Sınıflandırma Kural Çıkarımı, Yapay Sinir Ağları, Karınca Koloni Optimizasyonu.

RULE EXTRACTION FROM ARTIFICIAL NEURAL NETWORKS BY ANT COLONY OPTIMIZATION

Sinem KULLUK

Erciyes University, Graduate School of Natural and Applied Sciences
Ph. D. Thesis, April, 2009
Thesis Supervisors: Prof. Dr. Hüseyin YAPICI,
Assist. Prof. Lale ÖZBAKIR

ABSTRACT

Extracting classification rules from data is an important and challenging task of data mining applications. Many approaches have been proposed in the literature so far in order to develop effective algorithms for classification rule extraction. The topic is still attracting interest of researchers due to its central role in data mining research.

Artificial Neural Networks (ANNs) is one of the most widely used techniques in classification task of data mining. Although ANNs can achieve high classification accuracies, an important drawback of them is their lack of explanation capability. ANNs are black boxes and it is very difficult to understand how they learned and solved a problem. The main challenge in using neural networks in classification problems is to get explicit knowledge from these models.

For this purpose in this thesis, a knowledge extraction algorithm from trained ANNs for classification problems is presented. The proposed rule extraction algorithm actually works on the trained ANNs in order to discover the hidden knowledge which is available in the form of connection weights within ANN structure. The proposed algorithm is mainly based on a meta-heuristic which is known as Touring Ant Colony Optimization (TACO) with a two-step hierarchical structure. In the first step, a multilayer perceptron type neural network is trained and its weights are extracted. After obtaining the weights, in the second step TACO algorithm is applied to generate classification rules.

The proposed algorithm is experimentally analyzed and evaluated on 12 binary and n-ary benchmark data sets. These experimental studies and comparisons with some other classical and state-of-the art rule extraction algorithms shown that the proposed approach has a big potential to discover accurate and concise classification rules.

Keywords: Data Mining, Classification Rule Extraction, Artificial Neural Networks, Ant Colony Optimization.

İÇİNDEKİLER

KISALTMA VE SİMGELER

VM	: Veri madenciliği
YSA	: Yapay sinir ağları
TAKKO	: Tur atan karınca koloni optimizasyonu
VTBK	: Veritabanlarından bilgi keşfi
KKO	: Karınca koloni optimizasyonu
TAKKOYSA-sınıflandırıcısı	: Tur atan karınca koloni optimizasyonu ile yapay sinir ağları sınıflandırıcısı
ÇKA	: Çok katmanlı algılayıcı
EA	: Evrimsel algoritmalar
SZ	: Sürü zekası
DTN	: Durum tabanlı nedenleme
KK	: Kaba küme
BK	: Bulanık küme
GA	: Genetik algoritma
GP	: Genetik programlama
EP	: Evrimsel programlama
ES	: Evrimsel strateji
PSO	: Parça sürü optimizasyonu
SDA	: Stokastik difüzyon arama
tp	: Pozitif doğru
fp	: Pozitif yanlış
tn	: Negatif doğru
fn	: Negatif yanlış
DVM	: Destek vektör makineleri
ART	: Adaptif rezonans teori
AKA	: Arı kolonisi algoritması
KS	: Karınca sistemi
KKS	: Karınca koloni sistemi
SKKO	: Sürekli karınca koloni optimizasyonu
ITKKO	: Izgara tabanlı karınca koloni optimizasyonu
API	: Pachycondyla Apicalis optimizasyon algoritması
DT	: DecisionTable

TABLOLAR LİSTESİ

ŞEKİLLER LİSTESİ

1. BÖLÜM
GİRİŞ

1.1. Çalışmanın Kapsamı ve Önemi

Bilgisayar teknolojilerindeki hızlı gelişmelerle birlikte, üretilen sayısal bilgi miktarı da her geçen gün artmaktadır. Oluşan bu büyük veri yığınlarının yönetilmesi ve anlamlı sonuçların elde edilmesi ise büyük bir problem arz etmektedir. Bu problemle başa çıkmanın en etkin yolu ise veritabanlarından bilgi keşfidir (VTBK).

1960'lı yıllarda veri toplama ile başlayan veritabanı teknolojilerinin gelişim süreci, 1970'lerde veritabanları oluşturulması ile devam etmiştir. 1980'li yıllarda gelişen ilişkisel veritabanı yönetim sistemi, ileri veri modelleri ve uygulama kaynaklı veritabanı yönetim sistemleri 1999 ve 2000'li yıllarda yerini veri madenciliği ve veri ambarlarına bırakmıştır.

Veritabanlarından bilgi keşfi ve veri madenciliği terimleri çoğu zaman eş anlamlı olarak kullanılmaktadır. VTBK, veri içerisindeki geçerli, yeni, yararlı ve sonuç olarak anlaşılabilir örüntülerin çıkarılması sürecidir. Bunun yanında veri madenciliği terimi verideki kullanışlı örneklerin bulunmasını ifade eder. VM, verilerden modeller veya örnekler üretmek için veri analizleri ve keşif algoritmaları kullanmayı gerektirir. Dolayısıyla VM ile VTBK terimlerini eşanlamlı olarak değerlendirmek yanlış olur. Çünkü VTBK, veri madenciliğini de içine alan daha geniş bir kavramdır.

Veri madenciliği veya başka bir ifadeyle "veriden gizli tahminleyici bilginin keşfi", kullanıcıların büyük veri kümelerindeki en önemli bilgiye odaklanmalarına yardımcı olacak potansiyele sahip güçlü bir teknolojidir. Veri madenciliğinin temel amacı sadece doğru olan bilginin değil, aynı zamanda kullanıcı için anlaşılır ve ilginç olan bilginin de

keşfidir. Kümeleme, birliktelik kuralları, veri genelleme ve özetleme gibi birçok veri madenciliği algoritmaları arasında, gelecek veri nesnelerinin sınıflarını tahmin etmeyi amaçlayan "sınıflandırma" son yıllarda büyük ilgi çekmektedir.

Sınıflandırma, sınıf etiketi bilinmeyen nesnelerin sınıflarını tahmin etmekte elde edilen modeli kullanmak amacıyla, veri sınıf ve kavramlarını tanımlayan modeller veya fonksiyonlar kümesinin bulunması sürecidir [1]. Sınıflandırma, eğitim kümesinin analizine dayanan bir sınıflandırma modeli oluşturur. Sınıflandırmada, bir kural genel olarak keşfedilmiş bilgiyi "EĞER ….. O HALDE" önerme formunda şu şekilde gösterir: EĞER "durum(lar)" O HALDE "sınıf". Kuralın önde gelen (durumlar) kısmı tahminleyici değişkenlerin mantıksal kombinasyonlarını içerir ve kuralın sonra gelen (sınıf) kısmının tahminleyici değişkenleri, kuralın önde gelen kısmını sağlayan durumlar için tahmin edilen sınıfı içerir. Burada sınıflandırma kural çıkarımının temel amacı, veride gizli olan bilgiyi ortaya çıkarıp anlaşılır şekilde ifade etmek, daha önce bilinmeyen ilişkileri ortaya koymak, nedenleme ve tanımlama kabiliyeti sağlamaktır [2].

Tan vd. [3], sınıflandırma metotlarını kural tabanlı ve kural tabanlı olmayan metotlar olmak üzere ikiye ayırmıştır. Kural tabanlı metotlara C4.5, karar tabloları, ID3 gibi algoritmalar örnek verilebilir. Yapay sinir ağları ve destek vektör makineleri ise kural tabanlı olmayan metotlara örnektir. Kural-tabanlı sınıflandırma metotları veriden gizli bilgiyi doğrudan çıkarırlar ve kullanıcılar bu bilgileri kolaylıkla anlayabilirler. Kural tabanlı olmayan sınıflandırma metotları ise genellikle kural tabanlı sınıflandırma metotlarına göre daha doğru sonuçlar verirler fakat kara-kutu gibi davrandıklarından dolayı anlaşılırlık yönünden rekabetçi değillerdir.

Kural tabanlı olmayan sınıflandırma metotları grubuna giren yapay sinir ağları, sınıflandırmada en yaygın kullanılan tekniklerden biridir. Bir yapay sinir ağı, biyolojik sinir ağlarına dayanan matematiksel veya hesapsal bir modeldir. YSA, birbirine bağlı yapay nöronlar grubundan oluşur ve hesaplamada bağlantılı yaklaşım kullanarak bilgi dağıtır. Bir yapay sinir ağı birçok durumda, öğrenme sürecinde ağ boyunca dolaşan iç ve dış bilgiye göre yapısını değiştiren adaptif bir sistemdir. YSA, girdiler ile çıktılar arasındaki karmaşık ilişkileri modellemekte veya verideki örüntüleri bulmakta kullanılabilir.

YSA yöntemi, sınıflandırmada ve verinin tahmin edilmesinde oldukça doğru sonuçlar vermektedir. Bununla birlikte, yapay sinir ağlarının kara kutu gibi davranma ve kullanıcının anlamasına yardımcı olacak yüksek düzeyli kurallar keşfedememe gibi dezavantajları vardır [4]. Bunun sebebi, tipik YSA çözümlerinin, yorumlanması zor gerçek değerli parametrelerle karakterize edilen çok sayıda etkileşimli, doğrusal olmayan elemanlardan oluşmasıdır [5].

YSA'nın ürettiği sonucu açıklamak zordur çünkü sinir ağında gizli olan bilgi aktivasyon değerlerine ve nöron bağlantılarına dağıtılmıştır [6]. Bu nedenle birçok araştırmacı, yapay sinir ağlarının insanlar tarafından anlaşılır gösterimlerini bulmak için çeşitli algoritmalar geliştirme eğilimindedirler.

Yapay sinir ağlarından kurallar çıkaran algoritmalar, çıkarılan kurallar arasındaki ilişkiyi ve eğitilmiş YSA'ların içyapısını ortaya koyan sınıflandırmanın yarı şeffaflık boyutuna göre analizci, eğitimci ve seçici olmak üzere üç kategoriye ayrılmaktadır.

Analizci yaklaşımlar, kural çıkarımını ikili formda gizli ve çıktı birimler düzeyinde ele alırlar [7]. Gizli katmanların aktivasyon değerleri ve ağırlıklarını analiz ederek kurallar çıkarırlar.

Eğitimci yaklaşımlar, girdileri doğrudan çıktılara eşleştirmeye çalışırlar ve YSA'ları kara kutu olarak gösterirler. Kuralları çıkarırken sadece girdi ve çıktı aktivasyonlarından faydalanırlar [8]. Bu yaklaşımda çıkarılan kural sayısı ve kuralların yapısı ele alınan sinir ağının yapısından ve ağırlık sayısından bağımsızdır.

Son olarak seçici yaklaşımlar hem analizci, hem de eğitimci tekniklerin elemanlarını içermektedir. Literatürde yapay sinir ağlarından kurallar çıkarmak için birçok çalışma yapılmıştır. Bu alanda ilk çalışma bağlantıcı uzman sistemler üzerine çalışan Gallant [9] tarafından yapılmıştır. Gallant'ın çalışmasında her bir sinir ağı işlem elemanı, kavramsal bir birimi ifade etmektedir. Bu çalışmadan sonra çok çeşitli teknikler kullanılarak farklı kural çıkarım algoritmaları geliştirilmiştir. Bu çalışmaların birçoğu yapay sinir ağlarından kural çıkarmak için evrimsel algoritmaları kullanmaktadır.

Karıncaların yiyecek arama davranışı örnek alınarak Dorigo [10] tarafından geliştirilen karınca koloni optimizasyonu algoritması (KKO), günümüze kadar bir çok

optimizasyon problemine başarı ile uygulanmıştır. Parpinelli ve arkadaşları tarafından [11] geliştirilen "AntMiner" algoritması gibi birçok KKO temelli sınıflandırıcı veri madenciliğinin sınıflandırma fonksiyonunda oldukça kaliteli çözümler üretebilmektedir.

Karınca koloni optimizasyonu algoritmasının geniş bir uygulama alanına sahip olmasına rağmen, yapılan literatür araştırmasında KKO'nun eğitilmiş yapay sinir ağlarından kural çıkarımına herhangi bir uygulaması olmadığı gözlenmiştir. Bu açığı gidermek amacıyla tez çalışmasında, eğitilmiş yapay sinir ağlarından doğru ve anlaşılır sınıflandırma kuralları çıkarmak için karınca koloni algoritması temelli TAKKOYSA-sınıflandırıcısı geliştirilmiştir.

Önerilen algoritma eğitilmiş yapay sinir ağlarından sınıflandırma kuralları çıkarmak için ağırlık değerlerini kullanmakta; aktivasyon fonksiyonu üzerine herhangi bir tahmin yapmamaktadır. TAKKOYSA-sınıflandırıcısı ikili veri kümeleri üzerinde çalışmaktadır. Bunun nedeni ikili gösterim şekli ile yapay sinir ağı yapısındaki ağırlıkların ifade kolaylığıdır. İkili gösterim ile aktif hale gelen bağlantılar "1", aktif olmayan bağlantılar ise "0" değeri ile ifade edilerek, yapay sinir ağının çıktı fonksiyonunun hesaplanması mümkün olabilmektedir.

Geliştirilen TAKKOYSA-sınıflandırıcısında ilk olarak, yapay sinir ağları ikili veri kümeleri ağa gösterilerek eğitilmekte ve ağın hatasını en küçükleyen ağırlık değerleri elde edilmektedir. Ancak yapay sinir ağının eğitilmesinden elde edilen bu ağırlık değerleri, kullanıcının anlayabileceği herhangi bir bilgi sunmamaktadır.

Yapay sinir ağının bu ağırlık değerlerinde gizli olan bilgiyi kullanıcının anlayacağı forma dönüştürmekte karınca koloni optimizasyonu algoritmasından faydalanılmıştır. KKO ile ilgili ağırlık değerleri ve ikili veri kümeleri kullanılarak, yapay sinir ağının çıktı fonksiyonunu en iyileyen kurallar üretilmiş ve daha sonra bu kurallar indirgenerek dilsel ifadelere dönüştürülmüştür. Elde edilen kural kümelerinin performansları test veri kümesi üzerinde elde edilen doğruluklara ve ortalama kural sayılarına göre değerlendirilmiştir.

Önerilen algoritma, karınca koloni algoritmalarından sürekli optimizasyon problemleri için Hiroyasu ve arkadaşları [12] tarafından geliştirilen tur atan karınca koloni

optimizasyonu algoritmasına dayanmaktadır. TAKKOYSA-sınıflandırıcısında TAKKO algoritması, ağırlıkları ifade eden kuralların üretilmesinde kullanılmıştır.

Yapay sinir ağı modeli olarak ise sınıflandırma, tanıma ve genellemede oldukça yetenekli olan çok katmanlı algılayıcı modeli (ÇKA) kullanılmıştır. ÇKA modelinin seçilmesinin bir başka nedeni de katmanlar arası işlem elemanları bağlantıların tam olmasıdır. Çok katmanlı algılayıcı modeli kullanılarak ağın hatasını en küçükleyen ağırlık kümeleri elde edilmiştir.

Eğitilmiş YSA'dan TAKKO ile kural çıkarım algoritmasının en etkin parametrelerinin belirlenmesi için L-36 Taguchi deney tasarımı gerçekleştirilmiştir. Tam faktöryel tasarım yerine Taguchi tasarımı gerçekleştirilmesinin nedeni algoritmanın YSA eğitim ve kural üretim bölümlerinden gelen çok sayıda parametrenin olması ve parametre sayısıyla birlikte tasarım sayısının üstel olarak artmasıdır.

Geliştirilen algoritmanın performansı tahminleyici doğruluk ve kural sayısı olmak üzere iki performans ölçütü bakımından 12 referans sınıflandırma veri kümesi üzerinde klasik makine öğrenme algoritmaları ve literatürde mevcut olan diğer bazı kural tabanlı sınıflandırıcılar ile karşılaştırılmıştır. Ayrıca elde edilen sonuçlar arasında anlamlı bir farklılık olup olmadığını belirlemeye yönelik istatistiksel analizler gerçekleştirilmiştir.

1.2. Çalışmanın Amacı ve Yöntemi

Tez çalışmasının temel amacı yapay sinir ağlarının kara kutu problemini çözecek ve sınıflandırmada elde ettikleri yüksek doğrulukları, kullanıcıların anlayabileceği dilsel kurallara dönüştürecek bir modelleme ve çözümleme yöntemi geliştirmektir. Yeni yöntem, YSA'da gizli olan bilgiyi anlaşılır kurallara dönüştürmekte TAKKO algoritmasından yararlanmaktadır. Böylece, literatürde yer alan bir boşluğun doldurulması amaçlanmaktadır.

Yapay sinir ağlarından kural çıkarılması problemine çözüm üretebilmek için öncelikle kullanılacak veri ve kural gösterimine karar vermek gerekmektedir. Geliştirilen algoritma eğitilmiş yapay sinir ağlarından sınıflandırma kuralları çıkarmakta YSA'ların ağırlık değerlerinden faydalanmaktadır. Ağırlıkların aktif veya pasif olduğunu göstermenin en etkin yolu ise ikili gösterimdir. Bu gösterimde değişkenin "1" değerini alması ilgili nöronun aktif , "0" değerini alması ise pasif olduğunu ifade etmektedir. Bu

gösterimde ele alınan veri kümesinin ikili veya kesikli değişkenler içermesi gerekir. Sürekli değişkenler içeren veri kümelerinde ise ön işlem olarak kesiklendirme uygulanmalıdır. Böylece her kesiklendirme noktası bir ikili değişken ile ifade edilebilir.

Katmanlar arası tam bağlantı içermesi ve sınıflandırmada yaygın olarak kullanılması nedeniyle YSA modeli olarak çok katmanlı algılayıcı tercih edilmiştir. Eğitilmiş ÇKA'lardan elde edilen ağırlıklara göre ÇKA'nın çıktı fonksiyonunu en iyileyen girdi vektörü ağırlıkların nöronları aktif hale getirmelerine göre belirlenmiştir.

ÇKA çıktı fonksiyonunun en iyilenmesi probleminin bir optimizasyon problemi olması nedeniyle, bir çok optimizasyon probleminde oldukça iyi sonuçlar veren KKO algoritması ÇKA ağırlıklarından sınıflandırma kuralları üretmek için kullanılmıştır. Daha sonra üretilen kurallar indirgenerek dilsel ifadelere dönüştürülmüştür.

Çalışmanın ilk aşamasında TAKKO ile eğitilmiş YSA'lardan sınıflandırma kuralları çıkaracak bir algoritma oluşturulmuştur. İkinci aşamada ise geliştirilen algoritma Delphi programlama dili ile farklı veri kümelerine uygulanabilecek esneklikte kodlanmıştır.

Geliştirilen algoritmanın etkinliğinin analiz edilebilmesi için UCI makine öğrenme deposundan alınan 12 farklı referans sınıflandırma problemi üzerinde deneysel çalışmalar ve analizler gerçekleştirilmiştir. Çalışmanın son aşamasında ise algoritmanın performansı klasik ve yeni dönem sınıflandırıcılar ile karşılaştırılmıştır.

1.3. Çalışmanın Bölümleri

Tez çalışmasının ikinci bölümünde veri madenciliği ve sınıflandırma detaylı olarak incelenmiştir. Sınıflandırmada meta-sezgisel uygulamalar ile ilgili detaylı bir literatür yine aynı bölümde verilmiştir.

Üçüncü bölüm çalışmanın çıkış noktası olan yapay sinir ağları ile ilgili bilgiler içermektedir. YSA yapısı, modelleri, aktivasyon fonksiyonları gibi birçok konu bu bölümde detaylı olarak açıklanmıştır.

Dördüncü bölüm yapay sinir ağlarından kural çıkarımını ele almaktadır. Bu bölümde kural çıkarım algoritmaları sınıflandırılarak incelenmiş, algoritmalar ve yapıları

hakkında kısa bilgiler verilmiştir. Yine aynı bölümde YSA'dan kural çıkarımı ile ilgili detaylı bir literatür verilmiştir.

Beşinci bölümde KKO algoritmasının genel yapısı verildikten sonra KKO algoritmaları tanıtılmıştır. Geliştirilen algoritmada kullanılması nedeniyle TAKKO algoritması detaylı olarak incelenmiştir.

Altıncı bölümde, KKO algoritması ile eğitilmiş yapay sinir ağlarından sınıflandırma kuralları çıkarmak için geliştirilen TAKKOYSA-sınıflandırıcısının yapısı ve işleyişi hakkında bilgi verilmiştir. Algoritmanın adımları örneklerle detaylı olarak açıklanmıştır.

Yedinci bölümde, önerilen yöntemin çeşitli etkinlik ölçütlerine göre değerlendirilmesi ve analizleri verilmiştir. Bu bölümün sonunda geliştirilen algoritmanın etkinliği literatürde yer alan sınıflandırıcılar, klasik sınıflandırıcılar ve YSA'dan kural çıkarım algoritmaları ile karşılaştırılmıştır.

Son olarak sonuç ve öneriler bölümünde, tez çalışmasından elde edilen sonuçlar ve öneriler tartışılmştır.

Deneysel çalışma ve analizlere ilişkin ilave sonuçlar Ekler'de verilmiştir.

2. BÖLÜM
VERİ MADENCİLİĞİ

2.1. Giriş

Bu bölümde tez çalışması konusunun da yer aldığı veri madenciliği ele alınmıştır. İlk olarak veri tabanlarından bilgi keşfi ve veri madenciliği genel olarak açıklandıktan sonra VM ile ilgili olarak süreç, disiplinler arası ilişki, karşılaşılan problemler, fonksiyonları, dikkat edilmesi gereken hususlar ile model ve algoritmalarına değinilmiştir.

Tez çalışmasında geliştirilen algoritmanın bir sınıflandırma algoritması olmasından dolayı VM fonksiyonlarından sınıflandırma detaylı olarak açıklanmıştır. Son olarak, literatürde yer alan metasezgisel algoritmalar ile sınıflandırma uygulamalarına örnekler verilmiştir.

2.2. Veritabanlarından Bilgi Keşfi

Geleneksel veri analizi yöntemleri üretilen bilginin ve bu bilgilerin saklandığı veritabanlarının boyutunun artmasıyla birlikte verinin analiz edilmesinde ya yetersiz kalmakta ya da büyük zaman kaybına neden olmaktadır. Oluşan bu büyük veri yığınları ile başa çıkmakta veri tabanlarından bilgi keşfi son yıllarda yaygın olarak kullanılmakta ve büyük ilgi çekmektedir. VTBK ifadesi ilk olarak Piatetsky-Shapiro [13] tarafından 1989 yılında gerçekleştirilen ilk VTBK çalışma grubu toplantısında kullanılmış ve konuyla ilgili kavram ve tanımlamalar ortaya konulmuştur. Ayrıca veri madenciliği terimi de VTBK'nın bir bileşeni olarak aynı toplantıda tanımlanmıştır.

VTBK yapay zeka, veritabanları ve istatistik gibi bir çok bilimi birleştiren bir yapı olup bu bilimlerle birçok yönden rekabet etmektedir. Genel olarak veri tabanlarından bilgi keşfi, "veride mevcut olan doğru, yeni, kullanışlı ve sonuç olarak anlaşılır örüntülerin belirlenmesi" [14] olarak tanımlanır. Bilgi keşfi, gerçek hayat nesnelerini, kavramlarını

ve düzenliliklerini tanımlayan ifadeler üretir. Bu ifadeler, yeni hipotezlerin otomatik olarak üretilmesi ve doğrulanması sürecinden elde edilir. VTBK, veri analizinden ve veride mevcut örüntülerin keşfinden daha fazla anlamlar ifade eder. VTBK, verinin nasıl saklanması ve algoritmaların büyük veri kümelerine nasıl uyarlanması gerektiği, sonuçların nasıl yorumlanacağı ve görselleştirileceği, insan-makine etkileşiminin nasıl modelleneceği sorularının cevabını aramaktadır.

Fayyad vd.'nin [14] öne sürdüğü gibi VTBK, veride mevcut bilginin bulunması sürecinin tamamını kapsar. VM ise bu süreçteki merkezi adımdır ve verilerden modeller veya örnekler üretmek için veri analizleri ve keşif algoritmaları kullanmayı gerektirir. Ancak veri madenciliği ile ilgili bu tanım, VTBK topluluğunun tamamı tarafından kabul görmemekte, kimisi bunu kabul ederken kimisi VM ile VTBK'yı eş anlamlı olarak ele almaktadır.

VTBK süreci birkaç adımdan oluşan etkileşimli ve iteratif bir süreçtir. Bu süreç, uygulama alanının öğrenilmesi ile başlar ve uygulamanın amaçları doğrultusunda hedef veri seti seçilir. Daha sonra, gürültülü ve tutarsız verilerin çıkarıldığı veri temizleme ve ön işleme basamağı gelir. Gerekli durumlarda veri, madenciliğe uygun bir forma dönüştürülür. Dördüncü basamak olan veri madenciliği, zeki yöntemler aracılığıyla büyük miktarda veriden anlamlı bilgilerin çıkarılması sürecidir. Çıkarılan örüntüler, içlerinden yararlı olanların belirlenmesi için değerlendirilir. VTBK'nin son basamağı ise, elde edilen bilginin görüntüleme ve bilgi gösterimi yöntemleri ile kullanıcıya sunulmasıdır.

Fayyad'a göre VTBK sürecinde yer alan adımlar şöyledir [15];

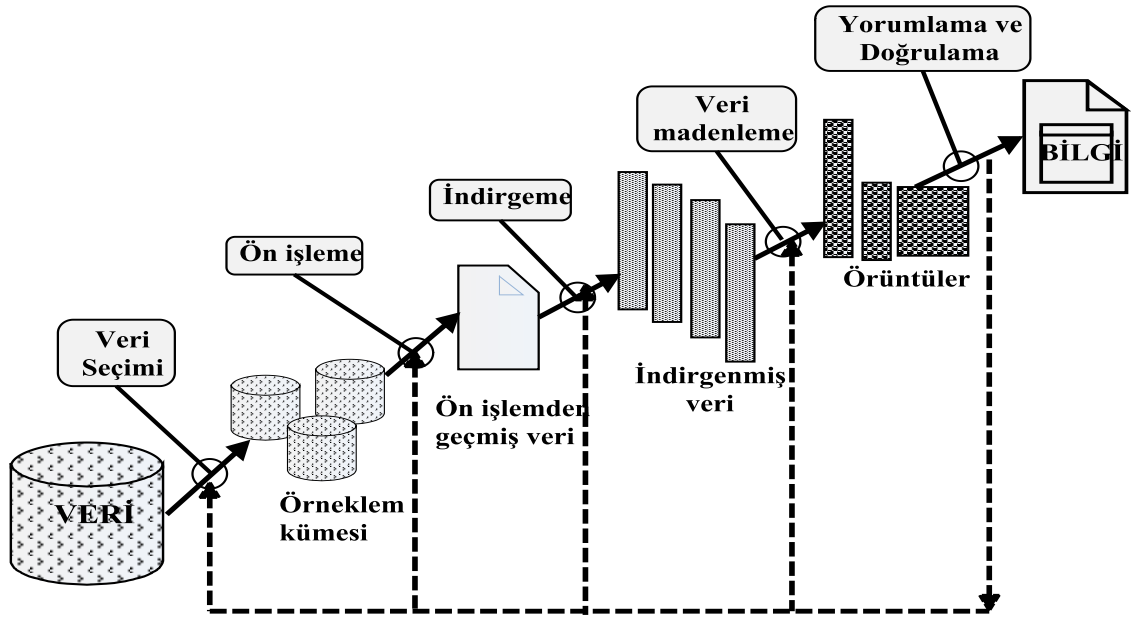
• ***Veri Seçimi***: Veri kümesinin birleştirilerek, sorguya uygun örneklem kümesinin elde edildiği adımdır.

• ***Veri Temizleme ve Önişleme***: Seçilen örneklemde yer alan hatalı örneklerin çıkarıldığı ve eksik nitelik değerlerinin değiştirildiği aşamadır. Bu aşama keşfedilen bilginin kalitesini arttırır.

• ***Veri İndirgeme***: Seçilen örneklemden ilgisiz niteliklerin atıldığı ve tekrarlı tutanakların ayıklandığı adımdır. Bu aşama seçilen veri madenciliği sorgusunun çalışma zamanını iyileştirir.

• ***Veri Madenciliği***: Verilen bir veri madenciliği sorgusunun (sınıflama, kümeleme, birliktelik analizi, vb.) işletilmesidir.

• ***Değerlendirme:*** Keşfedilen bilginin geçerlilik, yenilik, yararlılık ve basitlik gibi ölçütlere göre değerlendirilmesi aşamasıdır.



Şekil 2.1. Veritabanlarından bilgi keşfi süreci.

2.3. Veri Madenciliği Nedir?

Veri madenciliğinin bilgi endüstrisinde son yıllarda çektiği büyük ilginin temel sebebi büyük miktardaki verinin bulunabilirliği ve bu veriyi kullanışlı bilgiye çevirme ihtiyacıdır. 1990'lı yıllarda ortaya çıkan VM basit bir tanımla, büyük ölçekli veriler arasından bilgiye ulaşma, bilgiyi madenleme işidir. Ya da bir anlamda büyük veri yığınları içerisinden gelecek ile ilgili tahminde bulunabilmemizi sağlayabilecek bağıntıların bilgisayar programı kullanılarak aranmasıdır [1]. Bu noktada veri madenciliği kendi başına bir çözüm değil çözüme ulaşmak için verilecek karar sürecini

destekleyen, problemi çözmek için gerekli olan bilgileri sağlamaya yarayan bir araçtır. VM için yapılan diğer tanımlardan birkaçı şöyledir;

"*VM, önceden bilinmeyen, veri içinde gizli, anlamlı ve yararlı örüntülerin büyük ölçekli veritabanlarından otomatik biçimde elde edilmesini sağlayan, veritabanlarından bilgi keşfi süreci içinde bir adımdır*" [14].

"*VM, önceden bilinmeyen ve potansiyel olarak faydalı olabilecek, veri içinde gizli bilgilerin çıkarılmasıdır*" [16].

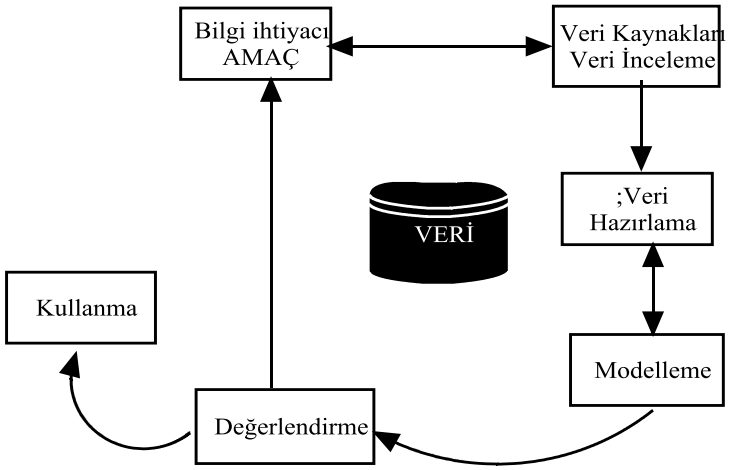
"*VM, büyük veri kümesi içinde saklı olan genel örüntülerin ve ilişkilerin bulunmasıdır*" [17].

"*VM, veri ambarlarında tutulan çok çeşitli verilere dayanarak daha önce keşfedilmemiş bilgileri ortaya çıkarmak, bunları karar vermek ve eylem planını gerçekleştirmek için kullanma sürecidir*" [18].

Veri madenciliği tanımları incelendiğinde, bu tanımlardan ortak olan unsurlardan ilki "çok fazla" miktarlarda verinin veri ambarlarında tutulması, ikincisi ise bu verilerden "anlamlı" bilgiler elde edilmesidir.

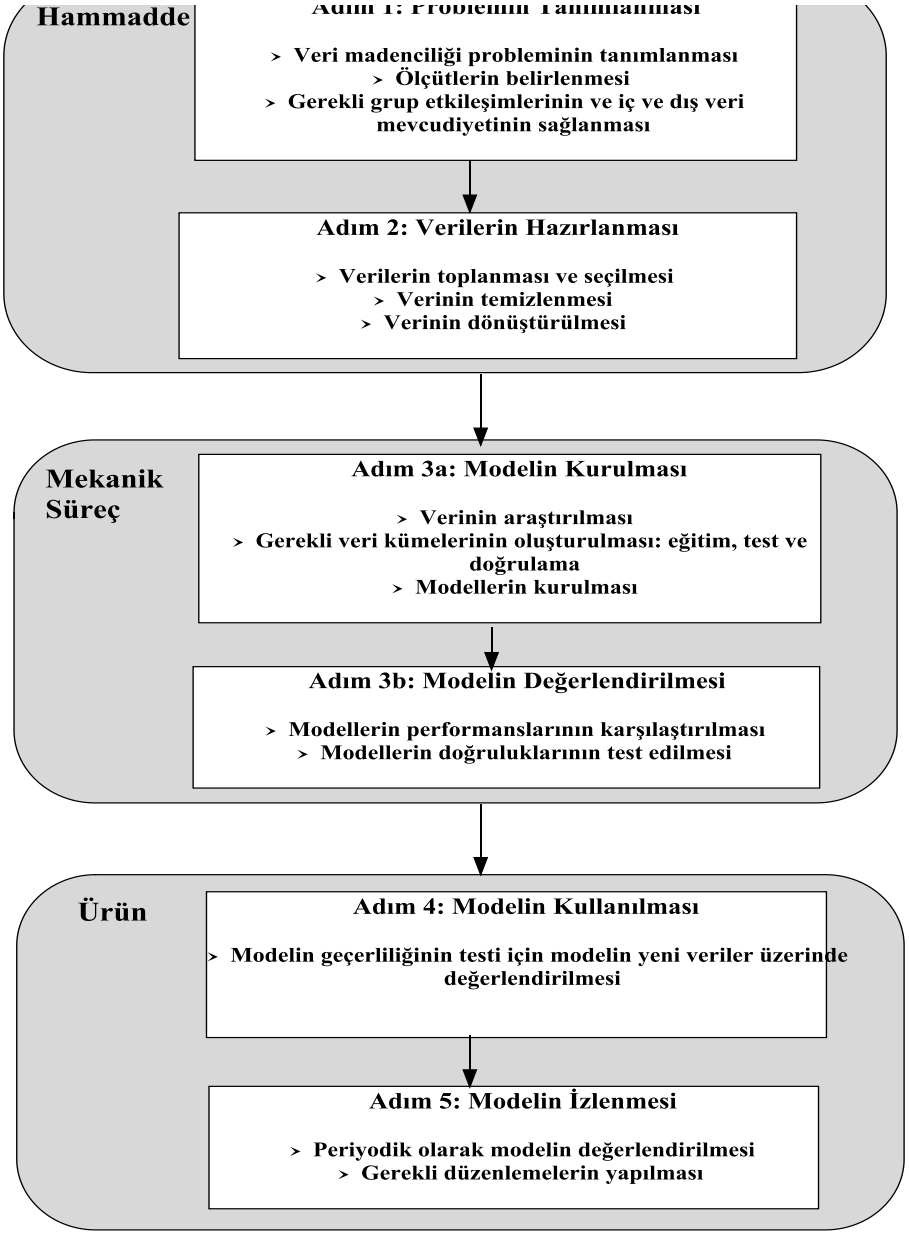
2.4. Veri Madenciliği Süreci

VM uygulamalarında başarının ilk anahtarı inceleme yapılan işin ve verilerin özelliklerinin tam olarak bilinmesidir. Bu bilgiler elde edildikten sonra veri madenciliği süreci etkin bir şekilde uygulanabilir. Şekil 2.2. veri madenciliği sürecini göstermektedir.



Şekil 2.2. Veri madenciliği süreci.

Şekil 2.2'de verilen VM sürecinden de görüldüğü gibi başarılı bir veri madenciliği uygulamasında izlenmesi gereken yol öncelikle problemin tanımlanması ve verilerin hazırlanmasıdır. Bu aşamada veriler hazırlandıktan sonra model kurulur ve değerlendirilir. Veriyi en iyi ifade eden model belirlendikten sonra ise model, kullanılarak değerlendirilir. Şekil 2.3. VM sürecinde yer alan adımları özetlemektedir.



Şekil 2.3. Veri madenciliği sürecinde yer alan adımlar.

Problemin Tanımlanması: Veri madenciliği çalışmalarında başarılı olmanın en önemli şartı, uygulamanın hangi amaçla yapılacağının açık bir şekilde tanımlanmasıdır. Amaç, açık bir dille ifade edilmeli, elde edilecek sonuçların başarı düzeylerinin nasıl ölçüleceği tanımlanmalıdır. Ayrıca yanlış tahminlerde katlanılacak olan maliyetlere ve doğru tahminlerde kazanılacak faydalara ilişkin tahminlere de bu aşamada yer verilmelidir.

Verilerin Hazırlanması: Veri madenciliğinin en önemli aşamalarından biri olan verilerin hazırlanması aşaması VM uygulamasının büyük bir bölümünü kapsamaktadır.

Bu aşamada sayısal bilginin iyi analiz edilmesi, veriler ile mevcut problem arasında ilişki olması gerektiği unutulmamalıdır. Verilerin hazırlanması aşaması kendi içerisinde toplama, birleştirme –temizleme ve dönüştürme adımlarından meydana gelmektedir.

Toplama: Tanımlanan problem için gerekli olduğu düşünülen verilerin ve bu verilerin toplanacağı veri kaynaklarının belirlenmesi adımıdır.

Birleştirme ve Temizleme: Bu adımda toplanan verilerde bulunan farklılıklar giderilmeye çalışılır. Hatalı veya analizin yanlış yönlenmesine sebep olabilecek verilerin temizlenmesine çalışılır. Genellikle yanlış veri girişinden veya bir kereye özgü bir olayın gerçekleşmesinden kaynaklanan verilerin, önemli bir uyarıcı enformasyon içerip içermediği kontrol edildikten sonra veri kümesinden çıkarılması tercih edilir.

Dönüştürme: Kullanılacak model ve algoritma çerçevesinde verilerin tanımlama veya gösterim şeklinin de değiştirilmesi gerekebilir.

Modelin Kurulması ve Değerlendirilmesi: Tanımlanan problem için en uygun modelin bulunabilmesi, olabildiğince çok sayıda modelin kurularak denenmesi ile mümkündür. Bu nedenle veri hazırlama ve model kurma aşamaları, en iyi olduğu düşünülen modele ulaşılıncaya kadar yinelenen bir süreçtir.

Model kuruluş süreci, danışmanlı ve danışmansız öğrenmenin kullanıldığı modellere göre farklılık göstermektedir. Örnekten öğrenme olarak da isimlendirilen danışmanlı öğrenmede, bir danışman tarafından ilgili sınıflar önceden belirlenen bir kritere göre ayrılarak, her sınıf için çeşitli örnekler verilir. Sistemin amacı verilen örneklerden hareket ederek her bir sınıfa ilişkin özelliklerin bulunması ve bu özelliklerin kural cümleleri ile ifade edilmesidir. Öğrenme süreci tamamlandığında, tanımlanan kural cümleleri verilen yeni örneklere uygulanır ve yeni örneklerin hangi sınıfa ait olduğu kurulan model tarafından belirlenir. Danışmansız öğrenmede, kümeleme analizinde olduğu gibi ilgili örneklerin gözlenmesi ve bu örneklerin özellikleri arasındaki benzerliklerden hareket ederek sınıfların tanımlanması amaçlanmaktadır.

Danışmanlı öğrenmede seçilen algoritmaya uygun olarak ilgili veriler hazırlandıktan sonra, ilk aşamada verinin bir kısmı modelin öğrenilmesi, diğer kısmı ise modelin

geçerliliğinin test edilmesi için ayrılır. Modelin öğrenilmesi, eğitim kümesi kullanılarak gerçekleştirildikten sonra, test kümesi ile modelin doğruluk derecesi belirlenir.

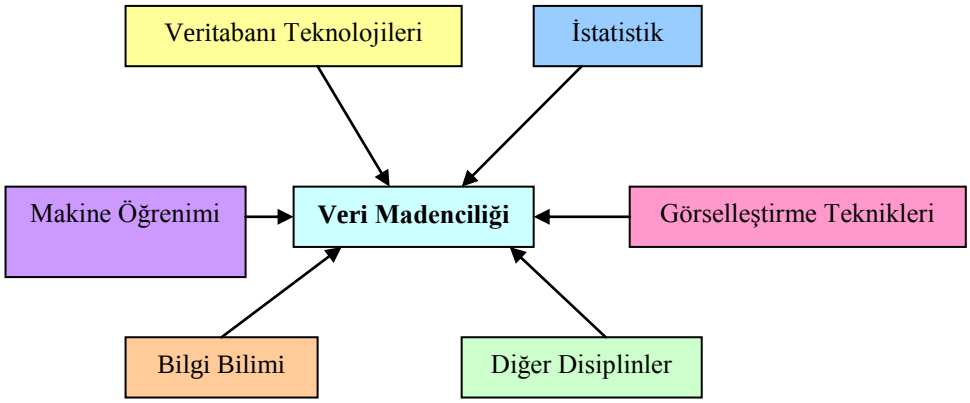
Modelin Kullanılması: Kurulan ve geçerliliği kabul edilen model doğrudan bir uygulama olabileceği gibi, bir başka uygulamanın alt parçası olarak da kullanılabilir. Kurulan modeller risk analizi, kredi değerlendirme, dolandırıcılık tespiti gibi işletme uygulamalarında doğrudan kullanılabileceği gibi, promosyon planlaması simülasyonuna entegre edilebilir veya tahmin edilen envanter düzeyleri yeniden sipariş noktasının altına düştüğünde, otomatik olarak sipariş verilmesini sağlayacak bir uygulamanın içine gömülebilir.

Modelin İzlenmesi: Zaman içerisinde bütün sistemlerin özelliklerinde ve dolayısıyla ürettikleri verilerde ortaya çıkan değişiklikler, kurulan modellerin sürekli olarak izlenmesini ve gerekiyorsa yeniden düzenlenmesini gerektirecektir. Tahmin edilen ve gözlenen değişkenler arasındaki farklılığı gösteren grafikler model sonuçlarının izlenmesinde kullanılan yararlı bir yöntemdir.

2.5. Veri Madenciliği ve Disiplinler Arası İlişki

Veri madenciliği, veritabanı teknolojisi, istatistik, makine öğrenme, yüksek-performanslı hesaplama, örnek kabulü, sinir ağları, veri görselleştirme, bilgi edinme, görüntü ve sinyal işleme ve uzaysal veri analizi gibi birçok farklı disiplinin kullandığı teknikleri kullanmaktadır. VM oluşturularak, veritabanlarından ilginç bilgi, benzerlikler veya yüksek düzeyli bilgiler çıkarılabilir ve farklı açılardan araştırılıp gösterilebilir. Keşfedilen bilgi, karar verme, süreç kontrol, bilgi yönetimi ve sorgu işlemeye uygulanabilir. Bununla birlikte veri madenciliği, veritabanı sistemlerindeki en önemli sınırlardan ve bilgi endüstrisindeki en umut vadeden disiplin gelişimlerinden biri olarak değerlendirilir.

VM, çok disiplinli bir yaklaşımdır ve birçok tekniği bünyesinde barındırmaktadır. Veri madenciliği ile makine öğrenimi, istatistik ve veri tabanı teknolojileri arasındaki yakın bağ kolaylıkla görülebilir. Bu disiplinler, veri içindeki ilginç birliktelikleri ve örüntüleri bulmayı amaçlar. Şekil 2.4. veri madenciliği ile disiplinler arası ilişkiyi göstermektedir.



Şekil 2.4. Veri madenciliği ile disiplinler arası ilişki.

Veri Madenciliği ile Makine Öğrenimi Arasındaki İlişki: Makine öğrenimi yöntemleri VM algoritmalarında kullanılan yöntemlerin çekirdeğini oluşturur. Makine öğreniminde kullanılan karar ağacı, kural tümevarımı yöntemleri pek çok VM algoritmasında kullanılmaktadır. Makine öğrenimi ile VM arasında benzerliklerin yanı sıra farklılıklar da göze çarpmaktadır. Öncelikle VM algoritmalarında kullanılan veri boyutu, makine öğreniminde kullanılan veri boyutuna nazaran çok büyüktür. Genellikle makine öğreniminde kullanılan veri boyu 100 ile 1000 arasında değişirken, VM algoritmaları milyonlarca gerçek hayat nesneleri üzerinde uğraşmaktadır ve bunlar karakteristiği boş, artık, eksik, gürültülü değerler olabilmektedir. Aynı zamanda VM algoritmaları, bilgi keşfetmeye uygun nesne niteliklerinin elde edilme sürecindeki karmaşıklıkla baş etmek zorundadır.

Veri Madenciliği ile İstatistik Arasındaki İlişki: VM ile istatistik arasındaki ilişkinin ana sebebi veri modelleme ve verideki gürültüyü azaltmadan kaynaklanmaktadır. Veri madenciliğini istatistiksel bir yöntemler serisi olarak görmek mümkündür. Veri madenciliğinde vurgulanan unsurlar istatistiğin tanımı içinde zaten yer almaktadır. İstatistik verilerin toplanması, sınıflandırılması, özetlenmesi, grafik ve tablolarla sunulması, analiz edilerek ana kütle hakkında anlamlı bilgiler elde edilmesi ve yorumlar yapılmasıdır.

Veri madenciliğinde ulaşılmak istenen amaç aslında istatistik biliminin amacı ile aynı doğrultudadır: verilerden bilgiyi keşfetmek. Zaten veri madenciliğinde kullanılan temel aracın istatistiksel yöntemler olduğu birçok tanımda ve uygulamada vurgulanmaktadır.

Her ikisinde de temel olan öğeler veri ve bilgidir. Bu nedenle birbiriyle oldukça örtüşen konulardır. Ancak veri madenciliği, geleneksel istatistikten birkaç yönde farklılık göstermektedir. Veri madenciliğinde amaç, kolaylıkla mantıksal kurallara ya da görsel sunumlara çevrilebilecek nitel modellerin çıkarılmasıdır. Bu noktada, veri madenciliği insan merkezlidir. VM, istatistik alanındaki birçok metodu kullanmasına rağmen, nesnelerin nitelik değerlerine bağlı çıkarım yapmakta bilinen istatistiksel metotlardan ayrılmaktadır. Örneğin ki-kare veya t-testi gibi istatistiksel test yöntemleri, birden fazla nitelik arasında korelasyon derecesini belirli bir güvenirlilik aralığında verebilmesine karşılık, belirli nitelik değerleri arasındaki ilişkinin derecesini açığa çıkaramazlar. Bu nedenle istatistiksel teknikler, veri madenciliği algoritmalarının uygulama kolaylığından dolayı VM kapsamı içerisindeki tekniklerden biri haline gelmiştir. Sonuç olarak istatistik ile veri madenciliği arasındaki ilişkinin ana sebebi veri modelleme ve verideki gürültüyü azaltmadan kaynaklanmaktadır.

Veri Madenciliği ile Veritabanı Teknolojileri Arasındaki İlişki: Veri madenciliği sorgularına girdi sağlamak amacıyla veri tabanları kullanılmaktadır. Veri tabanlarındaki sorgu cümlecikleri, veri madenciliği için gerekli veri kümesini elde etmek amacıyla kullanılmaktadır. Özellikle ilişkilendirme sorgusunda fazla miktarda veri tabanı sorgusu yapmak gerekmektedir. VM, veri tabanından farklıdır, çünkü veri tabanlarında var olan örüntüler için sorgular çalıştırılırken, veri madenciliğindeki sorgular genelde keşfe dayalı ve ortada olmayan örüntüleri bulmaya yöneliktir.

2.6. Veri Madenciliğinin Uygulama Alanları

Veri madenciliği günümüzde karar verme sürecinde ihtiyaç duyulan birçok alanda başarılı bir şekilde uygulanmaktadır. Uygulama alanlarına kısaca değinilecek olursa [1];

Pazarlama: Müşterilerin satın alma alışkanlıkları, demografik bilgileri, kampanya ürünleri belirleme, mevcut müşterileri kaybetmeden yeni müşteriler kazanma, market sepeti analizi, müşteri ilişkileri yönetimi ve satış tahmini alanları en yaygın veri madenciliği uygulama alanlarıdır.

Banka ve Sigortacılık: Farklı finansal göstergeler arasında korelasyon tespitinde, kredi kartı dolandırıcılıklarının tespitinde, kredi taleplerinin değerlendirilmesinde, kredi kartı harcamalarına göre müşteri profili belirlenmesinde, sigorta dolandırıcılıklarının

tespitinde ve yeni poliçe talep edecek müşterilerin tahmininde yoğun olarak kullanılmaktadır.

Biyoloji, Tıp ve Genetik: Bitki türleri ıslahı, gen haritasının analizi ve genetik hastalıkların tespiti, kanserli hücrelerin tespiti, yeni virüs türlerinin keşfi ve sınıflandırılması, fizyolojik parametrelerin analizi ve değerlendirilmesinde kullanılmaktadır.

Kimya: Yeni kimyasal moleküllerin keşfi ve sınıflandırılması, yem ve ilaç türlerinin keşfinde kullanılmaktadır.

Yüzey Analizi ve Coğrafi Bilgi Sistemleri: Bölgelerin coğrafi özelliklerine göre sınıflandırılmasında, kentlerde yerleşim yerleri belirlemede, kentlerde suç oranı, zenginlik-yoksulluk, köken belirleme ve kentlere yerleştirilecek posta kutusu, otomatik para makineleri, otobüs durakları gibi hizmetlerin konumlarının tespitinde kullanılmaktadır.

Görüntü Tanıma ve Robot Görüş Sistemleri: Çeşitli sensörler aracılığı ile tespit edilen görüntülerden yola çıkarak engel tanıma, yol tanıma, yüz tanıma ve parmak izi tanıma gibi tekniklerde kullanılmaktadır.

Uzay Bilimleri ve Teknolojisi: Gezegen yüzey şekilleri ve gezegen yerleşimleri, yeni galaksiler keşfi ve yıldızların konumlarına göre gruplandırılmasında kullanılmaktadır.

Meteoroloji ve Atmosfer Bilimleri: Bölgesel iklim, yağış haritaları oluşturma, hava tahminleri, ozon tabakası deliklerinin tespiti ve çeşitli okyanus hareketlerinin belirlenmesinde kullanılmaktadır.

Sosyal Bilimler ve Davranış Bilimleri: Kamuoyu yoklamaları inceleme, genel eğilim belirleme ve seçim öngörüleri oluşturmada kullanılmaktadır.

Metin Madenciliği: Çok büyük ve anlamsız metin yığınları arasından anlamlı ilişkiler elde etmekte kullanılmaktadır.

İş ve Elektronik Ticaret Verileri: Geri ofis, ön ofis ve ağ uygulamaları iş süreçleri sırasında geniş oranlarda veri üretirler. Bu veriyi karar verme mekanizmalarında etkin olarak kullanmak, ilgili ticari kuruluşun temel yapı taşlarından olmalıdır.

Bilimsel, Mühendislik ve Sağlık Bakım Verileri: Günümüzde bilimsel veriler, iş sahası verilerinden daha da karmaşık hale gelmişlerdir. Buna ek olarak, bilim adamları ve mühendisler uygulama sahası bilgilerini kullanarak simülasyon ve sistem kullanımının arttırılmasını amaçlamaktadırlar.

Web Verileri: Internet üzerindeki veriler hem hacim hem de karmaşıklık olarak hızla artmaktadır. Sadece düz metin ve resimden başka sürekli ve nümerik veriler de internet verileri arasında yer almaktadır.

2.7. Veri Madenciliğinde Karşılaşılan Problemler

Veri madenciliğinin büyük hacimli gerçek dünya verileriyle uğraşmasından dolayı birçok problemle karşılaşılmaktadır. Bu nedenle küçük veri kümeleriyle doğru çalışan sistemler büyük hacimli, eksik, gürültülü, boş değerli, artık veya dinamik verilerle yanlış çalışabilir. VM'de karşılaşılan problemler şu şekilde özetlenebilir [19].

Veritabanı Boyutu: Veri tabanı boyutları büyük bir hızla artmaktadır ve iki yönde genişlemektedir. Yatay boyutta nesnelerin özellik sayılarıyla, dikey boyutta ise nesnelerdeki örnek sayısıyla genişlemektedir. VM uygulamalarında ele alınan veri kümesinin büyük boyutlu olması bulunacak örüntüleri ne kadar iyi tanımlıyorsa, bu büyük küme ile uğraşma zorluğu da o kadar artmaktadır. Veri madenciliğinde veri hacminin büyüklüğünden kaynaklanan problemlerle başa çıkmanın bazı yolları şöyledir;

- Veri kümesinin yatay ve dikey boyutta indirgenmesi,
- VM yöntemlerinin sezgisel bir yaklaşımla arama uzayını taraması.

Gürültülü Veri: Verilerin toplanması veya girilmesi sırasında oluşan sistem dışı hatalar gürültü olarak ele alınır. Bu hataların neticesinde veri tabanlarında birçok niteliğin değeri yanlış olabilir. Veri madenciğinde kullanılan gerçek dünya verileri için bu sorun ciddi bir problemdir. Bu nedenle VM yöntemlerinin gürültülü verilere karşı daha az duyarlı olması yani gürültülü verilerin sistem tarafından tanınması ve ihmal edilmesi istenir. Quinlan [20], yaptığı bir çalışmada tümevarımsal karar ağaçlarında uygulanan yöntemler açısından gürültülü verinin yol açtığı problemleri araştırmıştır. Chan ve Wong [21], gürültünün etkisini azaltmak için istatistiğe dayanan yöntemler kullanmışlardır.

Boş Değerler: Veritabanındaki boş değerler, bilinmeyen ve uygulanamayan bir değer ifade eder. Lee [22], ilişkisel veritabanlarını genişletmek için boş değerleri üç gruba ayırmıştır;

- bilinmeyen,
- uygulanamaz,
- bilinmeyen ve uygulanamaz.

Bu ayrımdan genel olarak bilinmeyen değerler üzerinde daha çok çalışılmaktadır [23-27]. Boş değerlerle çalışabilmek için çeşitli yöntemler mevcuttur. Bunlar [20];

- Boş değerleri dikkate almama,
- Boş değerli kayıtlardaki boş değerlerin olası bir değerle doldurulması. Bunun için çeşitli yöntemler söz konusudur;
    - Boş değer yerine o nitelikteki en sık rastlanan bir değer veya ortalama bir değerin konulması,
    - Boş değer yerine varsayılan bir değer konulması,
    - Boş değerin bulunduğu örneğin diğer özelliklerine göre, boş değerin kendine en yakın değerle doldurulması.

Eksik Veri: Veri kümesindeki eksiklik yatay ve dikey boyutta mevcuttur. Bu eksiklikler şu şekilde olabilir;

- Yatay boyutta: Yatay boyuttaki eksiklik, veri kümesinde olması gereken nitelik veya niteliklerin olmamasıdır.
- Dikey boyutta: Dikey boyuttaki eksiklik veri kümesindeki kayıtların eksik olmasıdır.

Artık Veri: İstenilen sonucu elde etmek için kullanılan veri kümesindeki gereksiz nitelikler artık veri olarak isimlendirilir. Bu verileri elemek için geliştirilen algoritmalar ise özellik seçimi algoritmaları olarak bilinir. Özellik seçimi, arama uzayını daraltır ve sınıflandırma işleminin kalitesini artırır [28].

Dinamik Veri: İçeriği sürekli değişen veri tabanları dinamik veriler içerirler. Bu veritabanlarındaki içeriğin sürekli değişmesi veri madenciliği uygulamalarında sorunlar oluşturmaktadır. Bu sorunlardan bazıları şöyledir [29];

- Elde edilen veri madenciliği örüntülerinin dinamik verilerden hangisini ifade ettiğini tespit etmenin zorluğu, bu sonuçların zaman içinde eski üretilen sonuçlardan farkının tespiti ve gereken yerlerin güncellenme zorluğu,
- Veriler üzerinde VM algoritmaları çalıştırılırken, bu verilerin başka uygulamalar tarafından değişime açık olmaması,
- Veritabanı ve VM uygulamalarının aynı anda çalıştırılmasından dolayı performans düşüklüğü.

2.8. Veri Madenciliğinin Fonksiyonları

Veri madenciliği fonksiyonları, VM görevlerinde kullanılacak olan örüntü çeşitlerini tanımlamak için kullanılırlar. Genel olarak veri madenciliği fonksiyonları tanımlayıcı ve tahminleyici fonksiyonlar olmak üzere ikiye ayrılır. Tanımlayıcı veri madenciliğinde, veri kümesi kısa ve öz olarak tanımlanır ve verilerin ilginç genel özellikleri sunulur. Tahminleyici veri madenciliğinde ise bir veya daha fazla model(ler) kümesi oluşturulup mevcut verilerden sonuçlar çıkarılır ve yeni veri kümelerinin davranışları tahmin edilmeye çalışılır [30]. Veri madenciliği sistemlerinin farklı türlerde örnekler üzerinde madenleme işlemini gerçekleştirmesi ve kullanıcıların farklı ihtiyaç ve uygulamalarına cevap verebilmesi gerekmektedir. Veri madenciliğinin başlıca fonksiyonları şöyledir [1, 31];

- Sınıf tanımı,
- Birliktelik analizi,
- Sınıflandırma,
- Tahminleme,
- Kümeleme,
- Zaman serileri analizi,
- Aykırılık analizi,
- Evrimsel analiz.

Sınıf Tanımı: Veriler gösterdikleri ortak özelliklere göre genelleştirilmiş sınıflara ayrılabilirler. Sınıf tanımı, veri kümesinin kısa bir özetini sağlar ve diğerlerinden farklılığını ortaya koyar. Bir veri kümesinin özetlenmesine sınıf nitelendirmesi denilir. İki veya daha fazla veri kümesinin karşılaştırılmasına ise sınıf karşılaştırması ya da

ayrımı denilir. Sınıf tanımı, sadece özet özellikleri değil, aynı zamanda merkezi eğilim ölçüleri, dağılım ölçüleri gibi özellikleri de içerir.

Birliktelik Analizi: Birliktelik analizi, belirli bir veri kümesinde yüksek sıklıkta birlikte görülen nitelik değerlerine ait ilişkisel kuralların keşfedilmesidir. Temeli, bir veri kümesinde kendiliğinden sıklıkla gerçekleşen, birlikte ya da aynı süre içinde alınma, yapılma, oluşma gibi etkileri keşfetmeye dayanır. Bu yöntem bankacılık işlemlerinin analizinde veya sepet analizi tekniğinde yaygın olarak kullanılır.

Sınıflandırma: Sınıflandırma, sınıfı belirlenmemiş verilerin hangi sınıfa dahil edileceklerini tahmin etmek üzere, mevcut olan verileri farklı sınıflara atayabilmek için bazı fonksiyonlar içerir. Ele alınan veri kümesini çözümler ve verilerin özelliklerine göre sınıflar oluşturup, her bir sınıf için bir model kurar. Veritabanındaki her bir sınıfın daha iyi anlaşılması ve daha sonra elde edilen verilerin sınıflandırılması için sınıflandırma işlemi bir karar ağacı ya da sınıflandırma kuralları kümesi oluşturur. Makine öğrenimi, istatistik, veritabanı, sinir ağları, evrimsel algoritmalar ve bunlar gibi diğer alanlarda geliştirilmiş pek çok sınıflandırma yöntemi vardır. Geliştirilmiş olan yöntemler bir eğitim kümesinin analizine dayanmaktadır. Sınıflandırma sorgusuyla, bir kaydın önceden belirlenmiş bir sınıfa girmesi amaçlanmaktadır [32]. Bir kaydın önceden belirlenmiş bir sınıfa girebilmesi için öncelikle bir sınıflandırma algoritması ile eğitim verileri kullanılarak hangi sınıfların var olduğu ve bu sınıflara üyelik için bir kaydın hangi özelliklere sahip olması gerektiği belirlenmiş olmalıdır. Test verileri ile de bu öğrenmenin test işlemi yapılarak kurallar elde edilir.

Tahmin: Bu fonksiyon, bazı eksik verilerin olası değerlerini veya nesnelerin bir kümesinde kesin özelliklerin değer dağılımını ve seçilen nesnelere benzer verilerin kümesini temel alan değer dağılımını tahmin eder. Genellikle regresyon analizi, genelleştirilmiş doğrusal model, korelasyon analizi, karar ağaçları, genetik algoritmalar ve yapay sinir ağı modelleri nitelikli bir tahmin için kullanılan yöntemlerdir.

Kümeleme: Danışmansız öğrenme kategorisine giren kümeleme, özellikleri birbirine benzeyen nesnelerin değerlerinin toplanmasından oluşan bir veri kümesinde, verilerde mevcut kümeleri belirleme işidir. Benzerlik, uzmanlar veya kullanıcılar tarafından belirlenmiş uzaklık fonksiyonlarıyla tanımlanabilir. İyi bir kümeleme yöntemi, kümeler arası benzerliğin düşük ve küme içi benzerliğin yüksek olduğu nitelikli kümeler

meydana getirir. Veri madenciliği, büyük veritabanları ve çok boyutlu veri depoları için yüksek nitelikli ve hesaplanabilir kümeleme yöntemlerine odaklanır. Kümelemenin temel özellikleri şöyle sıralanabilir;

- Oluşacak küme sayısı belirsizdir,
- Kümeler hakkında bir ön bilgi olmayabilir,
- Küme sonuçları dinamiktir.

Zaman Serileri Analizi: Zaman serileri, benzer seriler ve ardışık örüntülerin madenciliği, periyodiklikler, eğilimler ve sapmaların araştırılması, ilginç özellikler ve kesin düzenlerin bulunması için zaman serilerinin geniş kümesini çözümler. Zaman serilerindeki örüntü belirli bir periyotta, belirli bir sıklıkla gerçekleşen olaylardır. Belirli sıklıkla tekrarlanan bu olaylar zaman serileriyle yapılan VM algoritmalarıyla keşfedilir.

Aykırılık Analizi: Bir veritabanı, tüm veri modelinin davranışını sergilemeyen veriler içeriyor olabilir. Bu tür veriler "aykırı" olarak adlandırılırlar. Birçok veri madenciliği tekniği aykırılıkları gürültü olarak ele alırlar. Buna rağmen hile tespiti gibi bazı uygulamalarda daha nadir oluşmuş olan olaylar sık oluşmuş olaylara göre daha ilginç ve önemli olabilirler. Aykırı verinin analizi, aykırılık analizi olarak adlandırılır. Aykırılıklar, istatistiksel testler kullanılarak belirlenebilir. Aykırılık analizinde iki yöntem söz konusudur. Bunlar;

İstatistik tabanlı yöntem: Dağılım analizi veya standart sapma hesabı gibi istatistik yöntemlerle aykırı olabilecek noktalar tespit edilir. Fakat çok büyük veri kümelerinde yoğun hesaplama gücü gerektirdikleri için performansları sınırlıdır.

Yoğunluk tabanlı yöntem: Bu yöntemde her noktanın çevresindeki komşuları ile olan yakınlığı hesaplanır. Yakınlık hesaplamada genellikle öklit uzaklığı kullanılsa da veri türüne göre yakınlık hesaplama yöntemi farklılık gösterebilir. Bu yöntemin temel prensibi, yeterince komşusu olmayan noktaları aykırı olarak seçmektir.

Evrimsel Analiz: Evrimsel analiz, zamanla davranışları değişen nesnelerin düzenlilik veya eğilimlerini ortaya çıkarmayı amaçlar. Tanımlama, ayrımlama, birliktelik analizi, sınıflama ve kümeleme metotlarını içerse de asıl amacı verinin zaman ile olan ilişkisini

ortaya çıkarmaktır. Bunun için zaman serileri, ardışıklık ve periyodiklik örüntüsü bulma, benzerlik analizi gibi yöntemleri kullanır.

2.9. Veri Madenciliğinde Dikkat Edilmesi Gereken Hususlar

Han ve Kamber'a [1] göre veri madenciliğinin önemli noktaları madenleme metodolojisi ve kullanıcı etkileşimi, performans ve veritabanı çeşitlerindeki farklılıklar olmak üzere üç grupta ele alınabilir. İlerleyen bölümlerde veri madenciliğinin önemli noktaları açıklanmaktadır.

2.9.1. Madenleme Metodolojisi ve Kullanıcı Etkileşimindeki Önemli Noktalar

Veri tabanındaki farklı tür verileri yakalayabilmek: Kullanıcıların veri tabanından istedikleri farklılaştıkça, aynı veri modeli üzerinde farklı veri madenciliği algoritmaları denenmeye başlanmıştır. Sınıflandırma, kümeleme, trend analizi, benzerlik analizi gibi yöntemler aynı veri kümelerine uygulanabilir.

Bilginin etkileşimli madenciliği: Veri madenciliği etkileşimli olunca, neyin keşfedileceğini bilmek zordur. Çok büyük boyutlu veri tabanları için, uygun örnekleme teknikleri öncelikle uygulanmalıdır. Etkileşimli madenleme işlemi OLAP'ın veri küpleri üzerinde sürekli olarak sorgulama yapabilmesi gibi bulunan örüntüler arasında sürekli sorgulama yapabilir.

Geçmiş bilgilerin dahil edilmesi: Geçmiş bilgiler keşif sürecini yönlendirmek için kullanılabilir ve keşfedilen örüntülerin farklı özetleme düzeylerinde kısa ve özlü ifadelerle tanımlanmasını sağlayabilir.

Veri madenciliği sorgulama dilleri: SQL gibi ilişkisel sorgu dilleri kullanıcıların verileri ortaya çıkarmasına olanak tanır. Veri madenciliği sorgulama dilleri ise, veriler arasında gizli kalmış örüntüleri ortaya çıkarmak için kullanıcıya sorgu olanağı sağlar.

Veri madenciliği sonuçlarının gösterimi ve görselleştirilmesi: Keşfedilen bilgi, yüksek düzeyli dillerde, görsel açıklamalarla veya diğer tanımlayıcı formlarla tanımlanmalıdır. Böylece bilgi kolayca anlaşılabilir ve kullanıcılar tarafından doğrudan kullanılabilir.

Gürültüyü ve eksik veriyi ele almak: Veritabanları genellikle hatalarla kirlenir, bu nedenle veritabanlarının tam olarak doğru veri içermediği varsayılır. Öznel veya ölçüye

dayalı nitelikler bazen hatalı sonuçlar verebilirler. Nitelik değerlerindeki veya sınıf bilgilerindeki hatalar, gürültü olarak adlandırılır. Gözlem olarak, üretilmiş kuralların toplam doğruluğu için sınıflanmış bilgiden gürültünün çıkarılması istenir.

Örüntü değerlendirme: Belirlenen ilginçlik ölçüm yöntemleri kullanılarak veri madenciliği ile bulunan verilerin ne kadar ilginç ve yararlı olduğu tespit edilir. Bunun için kullanıcı tanımlı eşik değerleri kullanılabilir.

2.9.2. Performans Ölçütleri

Veri madenciliği algoritmalarının etkinliği ve ölçeklenebilirliği: Büyük ölçekli veriler arasından bilgiye etkin şekilde erişimin gerçekleşmesi için, veri madenciliği algoritmaları kararlı ve etkin yapıda olmalıdır. Diğer bir deyişle, algoritmaların çalışma zamanları tahmin edilebilir ve geniş veriler düşünüldüğünde kabul edilebilir olmalıdır. Veritabanı bakış açısından değerlendirme yapıldığında, veri madenciliği teknikleri için anahtar kelimeler etkililik ve kararlılıktır.

Paralel ve dağıtılmış madenleme algoritmaları: Birçok veritabanının devasa hacmi, verinin geniş dağılımı ve bazı veri madenciliği metotlarının hesapsal karmaşıklığı paralel ve dağıtılmış veri madenciliği algoritmalarının geliştirilmesine neden olmuştur. Bu tip algoritmalar veriyi paralel işlem gören parçalara böler. Daha sonra parçaların sonuçları birleştirilir.

2.9.3. Veritabanı Çeşitlerindeki Farklılıklar

Verinin ilişkisel ve karmaşık türlerini ele alma: İlişkisel veritabanları ve veri depoları sıklıkla kullanıldığından dolayı, bu tip veriler için etkin ve etkili veri madenciliği sistemleri geliştirmek çok önemlidir. Veri tiplerinin karmaşıklığı ve veri madenciliğinin farklı amaçları ele alındığında, bir sistemden tüm veri tiplerinin madenlemesini beklemek gerçekçi olmaz. Verinin özel tipleri için özel veri madenciliği sistemleri geliştirilmelidir.

Heterojen veritabanlarından bilgi keşfi ve küresel bilgi sistemleri: Veri madenciliği, basit sorgulama sistemleri tarafından keşfedilemeyecek, çoklu heterojen veritabanlarındaki yüksek-düzeyli veri düzensizliklerini ortaya çıkarmaya yardımcı olabilir ve heterojen veritabanlarındaki uyumu ve bilgi değişimini geliştirebilir.

2.10. Veri Madenciliği Model ve Algoritmaları

Veri madenciliği modelleri gördükleri işlevlere göre;

- Kümeleme modelleri,
- Birliktelik kuralları ve ardışık zamanlı örüntüler,
- Sınıflandırma ve regresyon modelleri

olmak üzere üç gruba ayrılır. Bunlardan sınıflandırma ve regresyon modelleri tahmin edici; kümeleme ve birliktelik kuralları ile ardışık zamanlı örüntüler ise tanımlayıcı fonksiyonlara sahiptir.

2.10.1. Kümeleme Modelleri

Kümeleme, örnekler arasındaki yakınlık veya benzerliğin uygun niteliğine göre, örnek kümesinin belirli parça veya kümelere gruplanması süreci olarak tanımlanır [33]. Kümeleme ile birbirinden farklı olan ve birbirine benzeyen gruplar keşfedilmeye çalışılır. Kümelemede amaç, verileri alt kümelere ayırmaktır [34]. Kümelemede sınıflandırmadan farklı olarak, başlangıçta kümelerin ne olacağı veya hangi değişkenlerle verinin kümeleneceği bilinmez. Ele alınan konu ile ilgili uzman bir kişinin kümeleri yorumlaması gerekir. Kümeler belirlendikten sonra veri kümesi uygun bir şekilde parçalanmalıdır. Böylece bu kümeler daha sonra yeni verileri sınıflandırmakta kullanılabilir.

Kümelemede, benzerlikleri bulmak için çoğunlukla Manhattan ve Öklit uzaklık fonksiyonları kullanılır. Uzaklık fonksiyonu sonucunun yüksek bir değer olması az benzerlik olduğunu, düşük bir değer olması ise çok benzerlik olduğunu ifade eder. Veri kümeleri için uygulanacak uzaklık fonksiyonlarının verimi farklı olabilir. Bundan dolayı Manhattan ve Öklit haricindeki uzaklık fonksiyonları bazı veri kümeleri için daha uygun olabilir.

Kümeleme modellerinin özellikleri kısaca şöyle sıralanabilir;

- Danışmansız öğrenmedir,
- Kümelerin yapıları doğrudan veriden bulunmalıdır,

- Önceden tanımlanan sınıf ve sınıf etiketli öğrenme örnekleriyle çalışmamaktadır,
- Bir veri madenciliği fonksiyonudur,
- Veri dağılımını anlamaya yardımcı olur,
- Her bir kümenin özelliklerini izler.

Tipik bir örnek kümeleme faaliyeti beş adımdan oluşmaktadır. Bunlar [35];

1. Uygun örnek gösterimi
2. Nesneler arasında uygun uzaklık veya benzerlik ölçütü ile yakınlığın tanımı
3. Kümeleme
4. Veri soyutlama
5. Çıktının değerlendirilmesi

Kümeleme modellerinde yer alan algoritmalar Bölümleme Kümeleme Algoritması ve Hiyerarşik Kümeleme Algoritması olmak üzere ikiye ayrılır [29]. Bölümleme kümelemede, kümeler arasında ilişki bulunmaz, hiyerarşik kümelemede ise, her kümede veri örneklerini içerecek bir bağlantı kurulur. Bu algoritmaların sınıflandırılması aşağıda verilmiştir;

- Bölümleme Kümeleme Algoritması
    - K-ortalamalar
    - K-medoid (CLARANS)
    - Beklenen maksimizasyon algoritması
    - Danışmansız bayes
    - Mod bulma
    - Yoğunluk tabanlı yaklaşım
- Hiyerarşik Kümeleme Algoritması
    - Toplayıcı hiyerarşik kümeleme algoritması
    - Bölücü hiyerarşik kümeleme algoritması

Veri madenciliği amacı ile kullanılan kümeleme algoritmalarının aşağıdaki özellikleri taşıması gerekir [36];

- Ölçeklenebilir olmalı,

- Farklı veri tipleri ile başa çıkabilmeli,
- Göreli şekildeki kümeleri keşfedebilmeli,
- Girdi parametrelerini belirlemekte alan bilgisi için minimum ihtiyaç göstermeli,
- Aykırı değerlerle ve gürültü ile başa çıkabilmeli,
- Girdi kayıtlarının sırasına duyarsız olmalı,
- Yüksek boyutta olmalı,
- Kullanışlı ve yorumlanabilir olmalı.

2.10.2. Birliktelik Kuralları ve Ardışık Zamanlı Örüntüler

Birliktelik kuralları, büyük veri kümeleri arasındaki ilginç ilişkileri veya ilişki bağıntılarını yakalar. Birliktelik kurallarının matematiksel modeli Agrawal ve ark. [37] tarafından ortaya atılmıştır. Birliktelik kuralları analizinin uygulandığı genel bir alan sepet analizidir. Bu tür uygulamalar, müşterilerin sepetlerinde yer alan farklı ürünlerden yola çıkarak, müşterilerin alışveriş alışkanlıklarının yakalanabilmesini hedefler. Market stratejileri belirlenirken, birlikte satın alınan ürünlere ait sonuçlardan yararlanılabilir. Diğer veri madenciliği modellerinde olduğu gibi, birliktelik sorguları etkinlik, ölçeklenebilirlik, kullanılabilirlik ve anlaşılabilirlik gibi önemli ölçütleri sağlamalıdır.

Birliktelik kuralları ile çıkarılan kuralın, eldeki alternatifler kümesinin önemli bir kısmı tarafından desteklenmesi gerekir. Bu nedenle, birliktelik kuralı kullanıcı tarafından minimum eşik değeri belirlenmiş, güvenirlik ve destek ölçütlerini sağlayacak biçimde üretilir. Güvenirlik ölçütü, ilişkilendirme kuralının gücünü, destek ölçütü ise kuralda yer alan değişkenlerin geçiş sıklığını gösterir. Yüksek güvenirlik ve destek değerine sahip kurallara güçlü kurallar denilir. Birliktelik kuralları çıkarımı, büyük veritabanlarından güçlü ilişkilendirme örüntülerinin elde edilmesini sağlar.

Minimum güvenirlik ve destek ölçütlerini sağlayan birliktelik kuralları çıkarımı süreci iki adımdan oluşur [1];

1. *Sık rastlanan nesne kümelerinin bulunması:* Tanım olarak, her bir nesne kümesinin en az daha önce tanımlanmış minimum destek ölçüsü sıklığına sahip olması gerekir ki bu kümede yer alabilsin. Birliktelik sorgusu algoritmalarının performansını bu adım belirler.

2. *Sık rastlanan nesne kümelerinden güçlü birliktelik kuralları üretilmesi:* Tanım olarak bu kurallar minimum destek ve minimum güvenirlik düzeyini sağlamalıdır.

Sepet analizi, birliktelik kurallarının uygulandığı tek alan değildir. Birliktelik kuralları aşağıdaki kriterleri taşıyan çok değişik yollar ile sınıflandırılabilir;

- ***Kuralda ele alınan değişkenlerin tiplerine göre:*** Eğer bir kural nesnelerin varlık veya yokluğu arasındaki ilişkiyi ele alıyorsa, bu kural ikili birliktelik kuralıdır. Eğer bir kural nicel nesneler veya değişkenler arasındaki ilişkileri tanımlıyorsa, bu durumda bu kural nicel birliktelik kuralıdır.

- ***Kuralın içerdiği verinin boyutuna göre***: Bir birliktelik kuralındaki değişkenler veya nesneler bir boyutu temsil ediyorsa, o zaman kural tek boyutlu birliktelik kuralıdır. Eğer kural iki veya daha çok boyuta referans ediyorsa, bu durumda kural çok boyutlu bir birliktelik kuralıdır.

- ***Kural kümesinin içerdiği soyutlama düzeylerine göre***: Birliktelik kuralları analizi için bazı metotlar farklı soyutlama katmanlarında kurallar bulabilir.

- ***Birliktelik madenciliğinde çeşitli genişletmelere göre***: Korelasyon analizinin genişletilmesi örnek olarak verilebilir.

Birliktelik kuralları çıkarımını ele alan algoritmalara AIS, SETM ve Apriori algoritmaları verilebilir. Ardışık zamanlı örüntülerde, birliktelik kurallarından farklı olarak birbirleri ile ilişkisi olan ancak birbirini izleyen dönemlerde gerçekleşen ilişkilerin tanımlanması yapılır.

2.10.3. Sınıflandırma ve Regresyon Modelleri

Sınıflandırma, yüksek-düzeyli veri ile tanımlanan nesneler kümesi gibi karmaşık olayları, daha iyi kontrol veya ifade etmek için hizmet eden küçük ve tanımlayıcı birimlere, sınıflara, alt yapılara veya parçalara ayırarak ve mevcut bilgi temelinde yeni durumları bu sınıflardan birine atayarak ilerleyen temel zihinsel bir yetenektir [38]. Regresyon ise veriyi ilişkilendirmek, veriyle ilgili bağlantı kurmak için en yaygın

kullanılan yaklaşımlardan biridir. İstatistiksel regresyon yöntemleri genellikle, verilerin uydurulacağı fonksiyonu kullanıcının belirtmesini gerektirir [39].

Mevcut verilerden hareket ederek geleceğin tahmin edilmesinde faydalanılan ve veri madenciliği modelleri arasında en yaygın kullanıma sahip sınıflandırma ve regresyon modelleri arasındaki temel fark, tahmin edilen bağımlı değişkenin kategorik ve süreklilik gösteren bir değere sahip olmasıdır. Ancak çok terimli lojistik regresyon gibi kategorik değerlerin de tahmin edilmesine olanak sağlayan regresyon yöntemleriyle, her iki model giderek birbirine yaklaşmakta ve bunun sonucu olarak ele alınan problem için her iki teknikten de faydalanmak mümkün olmaktadır.

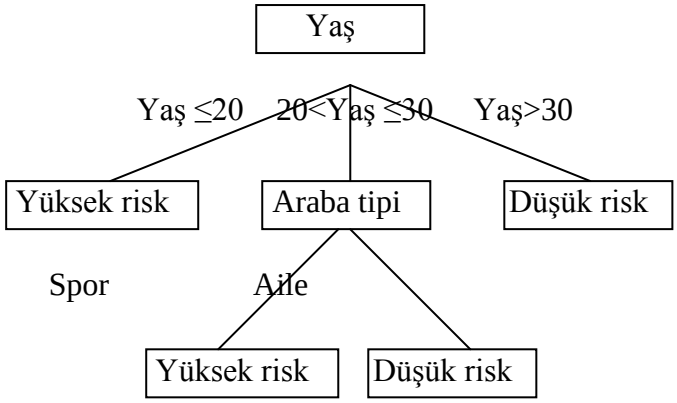
Sınıflandırma ve regresyon modellerinde kullanılan başlıca algoritmalar şöyle sıralanabilir;

- Karar ağaçları,
- Yapay sinir ağları
- Evrimsel Algoritmalar (EA)
- K-en yakın komşu
- Bayes sınıflandırıcılar
- Doğrusal regresyon
- Doğrusal olmayan regresyon
- Lojistik regresyon
- Sürü zekası (SZ) teknikleri
- Durum-tabanlı nedenleme (DTN)
- Kaba küme (KK) yaklaşımı
- Bulanık küme yaklaşımı (BK)

2.10.3.1. Karar Ağaçları

Karar ağaçları veri madenciliğinde genellikle veriyi tanımlamak, tahminde bulunmakta kullanılacak ağacı ve ağacın kurallarını indirgemekte kullanılan tekniklerdir. Karar ağacı akış diyagramına benzer bir ağaç yapısında olup, her bir dal bir testin sonucunu, yaprak düğümleri ise sınıfları temsil eder [1]. Bilinmeyen bir örneği sınıflandırmak için nitelik değerleri karar ağacı karşısında test edilir. Değerlendirme yapılırken yeni bir örnek, ağacın kök bölümünden girer. Kökte test edilen bu yeni örnek, test sonucuna

göre bir alt düğüme gönderilir. Bu süreç, yeni örnek herhangi bir yaprak düğüme ulaşana kadar devam eder. Ağacın belirli bir yaprağına gelen bütün örnekler aynı şekilde sınıflandırılırlar. Kökten her bir yaprağa giden sadece tek bir yol vardır. Bu yol, örnekleri sınıflandırmak için kullanılan bir kuralı tanımlamaktadır. Şekil 2.5. örnek bir karar ağacı yapısını göstermektedir.



Şekil 2.5. Örnek bir karar ağacı.

Bir karar ağacı, veri keşfine şu şekilde yardımcı olmaktadır [40];

- Temel özellik olarak korunan ve doğru bir özet sunan veriyi daha sıkıştırılmış bir hale sokarak, verinin hacmini azaltır.
- Veri iyi dağıtılmış sınıf nesneleri içersin veya içermesin karar ağacı keşifte bulunur, öyle ki sınıflar anlamlılık teorisi kavramında doğru bir şekilde yorumlanabilir.
- Veriyi bir ağaç formunda haritalar, böylece ağacın dallarından köküne doğru geri gidilerek tahmin değerleri üretilebilir. Bu değerler, yeni bir veri veya sorgunun çıktısını tahmin etmekte kullanılabilir.

Quinlan [41] tarafından geliştirilen ID3 ve C4.5 sıklıkla kullanılan karar ağacı algoritmalarıdır. Bu algoritmaların yanı sıra, doğruluğu artırmak için gürültülü verileri yansıtan dalları çıkaran budama algoritmaları da literatürde mevcuttur. Temel karar ağaçları iki gruba ayrılmaktadır. Bunlar [36];

- *Makine öğrenme topluluğundan sınıflandırıcılar*: ID3, C4.5, CART
- *Büyük veritabanları için sınıflandırıcılar*: SLIQ,SPRINT, SONAR, RainForest.

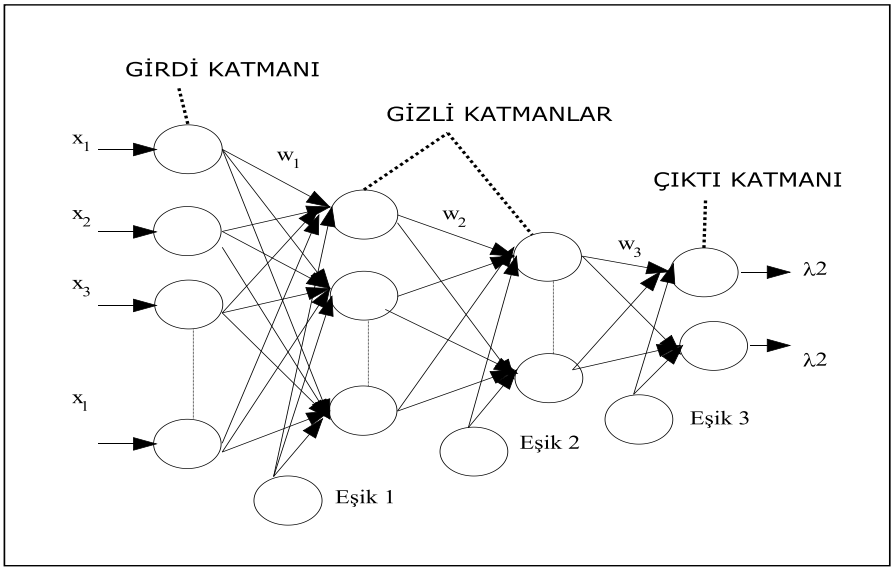
Karar ağaçlarının kolaylıkla sınıflandırma kurallarına dönüştürülebilmesi en büyük avantajlarından birisidir. Diğer avantajları ise şöyle sıralanabilir;

- Kuruluşlarının ucuz olması,
- Gürültülü verilerde etkili çalışması,
- Sınıfların ayırıcı özelliklerini keşfetmesi,
- Veritabanı sistemlerine kolayca entegre edilebilmeleri,
- Yorumlanmalarının kolay olması,
- Güvenilirliklerinin iyi olması.

2.10.3.2. Yapay Sinir Ağları

Tahmin ve sınıflandırma amacıyla kullanılan bir başka veri madenciliği tekniği yapay sinir ağlarıdır. Sinir ağları, bir ağa bağlı çok sayıda nöron bileşiminin matematiksel bir modelini sunarak, biyolojik sinir sistemini taklit etmeye çalışan sinyal işleme sistemleridir [42, 43]. YSA'da amaç, insan beyninin davranışlarını taklit etmektir [44]. Düğüm ve oklardan oluşan bir sinir ağında, düğümler nöronları, oklar ise sinyal akışının yönüyle beraber nöronlar arasındaki bağlantıları temsil eder. Nöronlar, giriş ve çıkış katmanlarında ve eğer varsa gizli katman(lar)da bulunur. Bir nöron önemli olarak belirlendiğinde, bu nöronla bağlantılı ağırlıklar değiştirilir. Bu da ilgili nöronun kendisiyle aynı düzeyde bulunan diğer nöronlara göre daha etkin olacağı anlamına gelir. Yapay sinir ağları nöronlar arasındaki ağırlıkların ayarlanması sayesinde öğrenirler. Şekil 2.6. basit bir yapay sinir ağı yapısını göstermektedir.

Sinir ağları, nöronlar arasındaki sinaptik bağlantıları ayarlamak suretiyle, girdi ile hedef çıktı eşleşecek şekilde eğitilir. Bu şekilde ağ, veri içinde gizli olan bilgiyi keşfeder. Sinir ağlarının gücü, şartlara ve çevreye uyum yeteneklerinden ve kendi kendilerini düzenleme kabiliyetlerinden ileri gelmektedir. Yapay sinir ağları ile ilgili daha detaylı bilgi Bölüm 3'de verilmiştir.



Şekil 2.6. Basit bir YSA yapısı.

Yapay sinir ağlarının veri madenciliği açısından olumlu ve olumsuz tarafları şöyledir;

- Çok geniş uygulama alanları mevcuttur,
- Karmaşık durumlarda daha iyi sonuçlar üretirler,
- Sürekli ve kategorik veriler üzerinde işlem yapabilirler,
- Elde ettikleri sonuçları açık bir şekilde ifade edemezler,
- Elde edilen sonucun en iyi sonuç olduğunun garantisi yoktur.

2.10.3.3. Evrimsel Algoritmalar

Doğal evrim sürecinden esinlenerek geliştirilen evrimsel algoritmalar, veri madenciliğinde sınıflandırma ve kural çıkarımında yaygın olarak kullanılmaktadır. Gradyan tabanlı tekniklerin aksine evrimsel algoritmalar, çoklu aday çözümlerinin performanslarını eşzamanlı olarak değerlendirerek zeki bir şekilde arama uzayını tararlar ve küresel en iyiye yaklaşırlar [45].

Genel olarak evrimsel algoritmalar, popülasyon ve seçim tabanlı olan, arama uzayında yeni bir arama noktası üreten genetik operatörlere sahip bütün algoritmaları içine alır. Bu algoritmalar genetik algoritmalar (GA) [46], genetik programlama (GP) [47], evrimsel programlama (EP) [48] ve evrimsel stratejilerdir (ES) [49]. Bu yaklaşımlar, kullandıkları operatörlere, uygulandıkları modellere, seçim metotlarına ve uygunluk

fonksiyonlarına göre birbirlerinden ayrılırlar. GA ve GP genetik seviyede evrimsel modellerdir. Evrimsel stratejilerde kullanılan optimizasyonda, popülasyondaki bireylerin yapıları en iyilenmeye çalışılır. Bireylerin çeşitli davranışsal özellikleri parametrik hale getirilir ve en iyilenme süresinde bu değerler geliştirilir. Evrimsel programlamada ise çeşitli türlerin davranışsal özelliklerinin adaptasyonu üzerinde durularak en üst düzey soyutlama kullanılır [50].

2.10.3.4. K-en Yakın Komşu

K-en yakın komşu, örnekleme yoluyla öğrenmeye dayanan en eski tekniklerden biridir. Bu teknikte tüm örneklemler n boyutlu uzayda saklanır [36]. Algoritma, bilinmeyen bir örneklemin hangi sınıfa dahil olduğunu belirlemek için örüntü uzayını araştırarak bilinmeyen örnekleme en yakın olan k örneklemi bulur. Yakınlık, genellikle öklit uzaklığı ile tanımlanır. Hedef fonksiyon kesikli veya reel değerli olabilir. Daha sonra, bilinmeyen örneklem, k en yakın komşu içinden en çok benzediği sınıfa atanır. Burada k harfi araştırılan komşuların sayısıdır. 5-en yakın komşuluğunda, 5 komşuya ve 1-en yakın komşuluğunda 1 komşuya bakılır [1].

K-en yakın komşu algoritması, aynı zamanda, bilinmeyen örneklem için bir gerçek değerin tahmininde de kullanılabilir. K-en yakın komşu algoritmasında uzaklık fonksiyonu ve dahil edilecek komşu sayısı parametresi olan k'nın değerleri ile ilgili kararlar en önemli seçimlerdir [51]. K-en yakın komşu modeli, özellikle çok fazla açıklayıcı değişken olduğu durumda, çok fazla hesaplama yükü getirmesinden dolayı dezavantaja sahiptir. Böyle bir durumda komşuluklar alakasız noktalarda çıkabilir ve anlamlı sonuçlar elde edilemeyebilir.

2.10.3.5. Bayes Sınıflandırıcılar

Bayes sınıflandırıcılar istatistiksel sınıflandırıcılardır. Özel bir sınıfa ait olarak verilen bir olasılık gibi, sınıf üyeliklerine ait olasılıkları önceden söyleyebilirler [1]. Bunun yanı sıra büyük veri tabanları üzerinde işlem yapan bayes sınıflandırıcılar hız ve doğruluk yönünden oldukça yüksek performansa sahiptirler. Bayes sınıflandırıcılar, verinin olasılık dağılımı verildiğinde en düşük hata oranıyla etkin bir şekilde çalışabilirler [52].

Naive bayes, verilen bir sınıf üzerindeki bir özelliğe ait değerin etkisinin diğer özelliğe ait değerlerden bağımsız olduğunu farz eder. Bu kabul, sınıf koşullu bağımsızlık olarak

adlandırılır. Bu ise gerekli hesaplamaları basitleştirir, anlaşılırlığı kolaylaştırır ve doğal olarak saf yani “naive” kelimesi ile ifade edilir.

Bayes sınıflandırıcılar verinin tek bir kez taranmasını gerektiren basit sınıflandırıcılardır. Bu nedenle büyük veri yığınlarında yüksek doğruluk ve hız elde edebilirler. Performansları karar ağaçları ve sinir ağları ile rekabet edebilecek ölçüdedir [36].

2.10.3.6. Doğrusal Regresyon

Regresyon analizi istatistiksel bir teknik olup, bir veya daha çok değişkenin başka değişkenler cinsinden tahmin edilmesini sağlayan ilişkilerin bulunmasına yardımcı olur. Regresyon analizinin değişik türleri vardır. Doğrusal regresyonda veri, düz bir doğru kullanılarak modellenir. Doğrusal regresyon, regresyonun en basit halidir. İki değişkenli doğrusal regresyon, rastsal bir değişken olan Y’yi (yanıt değişkeni) diğer bir rastsal değişken olan X’in (tahminleyici değişken) doğrusal bir fonksiyonu olarak modeller [1]. Fonksiyon şu şekildedir;

$$Y = \alpha + \beta x \qquad (2.1.)$$

Burada yanıt değişkeni Y’nin varyansı sabit olarak kabul edilir. α ve β, sırasıyla Y-kesişim ve doğrunun eğimini sağlayan regresyon sabitleridir. Bu sabitler, gerçek veri ve tahmin edilen doğru arasındaki hatayı en küçükleyen, en küçük kareler yöntemi ile çözülebilir. Doğrusal regresyon, bağımlı ve bağımsız tüm değişkenler nümerik olduğunda seçilecek en doğal yöntemdir [53].

2.10.3.7. Doğrusal Olmayan Regresyon

Veri doğrusal bir bağımlılık göstermiyorsa, yanıt değişkeni ve tahminleyici değişkenler ancak çok terimli bir fonksiyonla modellenebiliyorsa bu durumda doğrusal olmayan regresyon kullanılır. Çok terimli regresyon, temel doğrusal modele çok terimli ifadeler ekleyerek modellenir. Doğrusal olmayan regresyonda değişkenlere dönüşümler uygulanarak, doğrusal olmayan model doğrusal bir model haline dönüştürülür ve bu doğrusal model daha sonra en küçük kareler yöntemi ile çözülür.

2.10.3.8. Lojistik Regresyon

Lojistik regresyon, sınıflandırma analizlerinde yaygın olarak kullanılan doğrusal regresyonun genelleştirilmiş halidir. Lojistik regresyonda, genelleştirilmiş doğrusal modeller, kategorik bağımlı değişkenlerin tahmin edilmesinde kullanılır. Lojistik regresyon ilk olarak evet/hayır, 0/1 gibi ikili değişkenlerin bazen de çok sınıflı değişkenlerin tahmin edilmesinde kullanılmıştır [54].

Çok değişkenli normal dağılım varsayımına ihtiyaç duymadığından, lojistik regresyon bu tür uygulamalardan üstünlük sağlamaktadır. Ayrıca lojistik regresyonun sınıf üyeliğine ilişkin olasılıkları belirleme özelliği de vardır. Lojistik regresyonun varsayımlarından biri, doğrusal olasılık fonksiyonunun, hata terimlerinin dağılımının lojistik dağılıma uymasıdır.

Lojistik regresyon güçlü bir modelleme aracı olmasına rağmen, yanıt değişkeninin tahminleyici değişkenler katsayılarında doğrusal olduğunu farz eder. Ayrıca, modeli kuran kişi, veri ve veri analizi ile ilgili deneyimlerine göre doğru girdileri seçmeli ve bunların yanıt değişkeni ile fonksiyonel ilişkilerini belirlemelidir.

2.10.3.9. Sürü Zekası

SZ, özerk yapıdaki basit bireyler grubunun kolektif bir zeka geliştirmesi olarak tanımlanır [55]. SZ tanımı ilk olarak 1989 yılında Gerardo Beni ve Jing Wang [56] tarafından hücresel robotik sistemler kavramı içinde kullanılmıştır. SZ, merkezi olmayan, kendi kendini yöneten sistemlerde toplu davranış çalışmasına dayanmaktadır. Sürü zekası sistemleri genel olarak, birbirleriyle ve çevreleriyle etkileşen basit ajanlar topluluğundan oluşur. Birey olarak ajanların nasıl davranacağını dikte eden merkezileşmiş bir kontrol yapısı yoktur. Bu ajanlar arasındaki yerel etkileşimler çoğunlukla küresel davranışın ortaya çıkışını gösterir. Bu tip sistemlere örnekler doğada mevcuttur. Bunlara örnek olarak karınca kolonileri, kuş sürüleri, hayvan sürüleri, bakteri küfleri, arı kolonileri ve balık sürüsü vb. verilebilir.

Sürü zekası “stigmergy”, yani “etkileşim” ve “self-organization” yani “kendi kendine organizasyon” olmak üzere iki mekanizma üzerine kuruludur. Etkileşim, sürünün iletişim kurmasına ve sürüdeki elamanların birbirlerinin hareketlerini düzenlemelerine yardımcı olur. Kendi kendine organizasyon ise, sürünün herhangi bir plan olmadan

sonuç üretebilmesini, esnek, sağlam ve merkezi bir yönetim birimi olmadan yapılanmasını sağlar.

Sürü zekası, karınca koloni optimizasyonu [10], parça sürü optimizasyonu (PSO) [57] ve stokastik difüzyon arama (SDA) [58] olmak üzere farklı algoritmalar içermektedir. Bu algoritmalar optimizasyonda başarı ile kullanılmaktadır. Günümüzde sürü zekası teknikleri veri madenciliği alanında da kullanılmaya başlanmıştır ve yapılan uygulamalar bu tekniklerin sınıflandırmada iyi sonuçlar elde edebildiğini göstermektedir.

2.10.3.10. Durum Tabanlı Nedenleme

İlk olarak Schank [59] tarafından ortaya atılan DTN sınıflandırıcıları, örnek tabanlı sınıflandırıcılardır [1]. DTN, deneyimle öğrenir ve verilen problemle daha önce karşılaşılan problemlerin benzerlik ve farklılıklarından yararlanır. Eğitim örneklerini öklit uzayında noktalar olarak saklayan en yakın komşu sınıflandırıcılarından farklı olarak, DTN tarafından saklanan örnekler veya durumlar karmaşık sembolik tanımlamalardır. Yapay sinir ağlarından farklı olarak ise durum tabanlı nedenlemeye dayalı sınıflandırıcılar genelleme yapmazlar. Durum tabanlı nedenleme sınıflandırıcıları şunları içerir [60];

- Yeni istekleri karşılamak için eski çözümleri uyarlamak,
- Yeni durumları açıklamak veya yeni çözümleri ispatlamak için eski durumları kullanmak,
- Yeni durumları yorumlamak için önceki durumları nedenlemek.

Sınıflandırılacak yeni bir durum ele alındığında, bir durum tabanlı nedenleyici ilk olarak belirli bir eğitim durumunun mevcut olup olmadığını kontrol eder. Eğer bir eğitim durumu mevcutsa, ilişik bir çözüm o duruma geri gönderilir. Eğer hiçbir belirli durum bulunmazsa, bu durumda durum-tabanlı nedenleyici yeni durumun bileşenleriyle aynı bileşenlere sahip eğitim durumlarını arar. Kavramsal olarak bu eğitim durumları yeni durumun komşuları olarak düşünülebilir. Durum tabanlı nedenleyici yeni duruma bir çözüm önermek için komşu eğitim durumlarının çözümlerini birleştirmeye çalışır. Çözümlerde uyuşmazlık ortaya çıkarsa bu durumda yeni çözümler için yeniden arama

yapmak gerekebilir. Durum tabanlı nedenleyici, uygun bileşik bir çözüm bulmak için geçmiş bilgileri ve problem çözme stratejilerini kullanabilir.

DTN'ye yeni yaklaşımlar iyi bir benzerlik ölçüsünün bulunması, eğitim durumlarını indekslemek için etkin tekniklerin geliştirilmesi ve çözümlerin birleştirilmesini içerir. Mevcut bazı durum seçim metotları şunlardır [61]; özetlenmiş en yakın komşu, durum genişletme veya budama ile örnek tabanlı öğrenme (IB3 gibi), nitelik ağırlıklandırma ile örnek tabanlı öğrenme (IB4 gibi).

2.10.3.11. Kaba Küme Yaklaşımı

İlk olarak Pawlak [62] tarafından ortaya atılan KK yaklaşımı sınıflandırmada kesin olmayan, gürültülü verideki yapısal ilişkileri bulmakta kullanılır ve kesikli değerli değişkenlere uygulanır. Kaba küme yaklaşımı, klasik kümedeki, kümenin yalnızca elemanları ile tanımlandığı ve kümenin elemanları hakkında ilave hiçbir bilginin bulunmadığı yaklaşımın aksine, bir kümenin tanımlanması için başlangıçta uzay hakkında bazı bilgilere gereksinim olduğu varsayımına dayanır. KK yaklaşımının temelini ayırt edilememe ilişkisi oluşturur. Bilginin temelini oluşturan ve aynı nesnelerin kümesi olan kümeye "elemanter" küme denir. Elemanter kümelerin herhangi bir birleşimi ise "kesin" küme olarak adlandırılır, aksi halde bir küme "kaba" olarak ifade edilir. Her kaba kümenin kesinlikle kümenin kendisinin veya tümleyen kümesinin elemanları olarak sınıflandırılamayan elemanları vardır, bunlara "sınır hattı elemanları" denir [63].

Kaba küme yaklaşımı, ele alınan bir eğitim verisinde denklik sınıflarının kurulumuna dayanır. Bir denklik sınıfını oluşturan tüm veri örnekleri ayırt edilemez niteliktedir. Verilen bir sınıf için kaba küme tanımı iki küme ile tahmin edilir, bunlar "alt yaklaşım" ve "üst yaklaşım"dır. Alt yaklaşım kesin olarak sınıfa ait olan tüm nesnelerden oluşur. Üst yaklaşım ise sınıfa ait olması olası bütün nesneleri içerir. Alt ve üst yaklaşımlar arasındaki fark sınır bölgesini oluşturur [64]. Karar kuralları her bir sınır için oluşturulur. Genel olarak kaba küme yaklaşımında, kuralları göstermekte karar tabloları kullanılır [1].

KK yaklaşımı kullanılarak çözülebilen ana problemler; özellik değerleri cinsinden nesnelerin kümesinin tanımı, özellikler arasındaki tam veya kısmi bağımlılıkların

belirlenmesi, özelliklerin indirgenmesi, özelliklerin öneminin ortaya konulması ve karar kurallarının oluşturulmasıdır [62].

2.10.3.12. Bulanık Küme Yaklaşımı

Zadeh [65] tarafından ortaya atılan ve bulanık mantığa dayanan bulanık küme yaklaşımı, belirsizlik ile ilgilenir. Bulanık mantık, karmaşık, iyi tanımlanmamış ya da matematiksel olarak kolay analiz edilemeyen sistemlerin davranışlarını tanımlamak için hemen hemen doğru ve etkin yöntemler sunar. Diğer bir deyişle, bulanık mantık, belirsiz ve kesin olmayan bilginin kullanılması için bir platform oluşturur. Kategoriler arasında kesin bir ayrımdan ziyade, 0-1 arasında bir doğruluk değeri kullanırlar.

Bulanık küme yaklaşımları, veri madenciliği uygulamalarında özellikle sınıflandırma amacıyla kullanılmaktadır. Ancak sürekli değişkenler için keskin sınırlarının olması sınıflandırmada önemli bir dezavantajdır. Bununla birlikte yüksek düzeyde bir soyutlama sağlamaları sınıflandırmada bulanık küme yaklaşımına avantaj sağlamaktadır [1].

2.11. Sınıflandırma

2.11.1. Sınıflandırma Kuralları

Sınıflandırma, sınıf etiketi bilinmeyen nesnelerin sınıflarını tahmin etmek için elde edilen modeli kullanmak amacıyla, veri sınıf ve kavramlarını tanımlayan modeller veya fonksiyonlar kümesinin bulunması sürecidir [1]. Sınıflandırma işlemi genellikle veritabanlarından sınıflandırma modeli oluşturmak için danışmanlı öğrenme metotlarını kullanır. Çıktı sınıfı bilinen bir örnekler kümesi verildiğinde, sınıflandırmanın amacı değişkenler ve sınıflar arasındaki gizli ilişkilerin keşfedilmesidir [66]. Sınıflandırmada karar sınırları, farklı sınıflara ait örnekleri ayırt etmek için oluşturulur [36].

Sınıflandırma kuralları veri madenciliği uygulamalarında, çıktının gösterimi için en çok tercih edilen yöntemlerden biridir. Bunun nedeni kullanıcının kuralları anlamasının ve yorumlamasının kolay olmasıdır. Karar ağaçlarının düğümlerdeki testleri gibi bir kuralın önde gelen kısmı test serilerini içerir, sonra gelen kısmı ise o kuralca kapsanan sınıf veya sınıfları ifade eder [53]. Kurallar genellikle bilginin ifade edilmesinde kullanılır ve kullanıcılar tarafından kolayca anlaşılabilir.

Sınıflandırma geleneksel analizlere göre bazı avantajlara sahiptir ve bu avantajlar şu şekilde özetlenebilir [67];

- Çözüm zamanı daha kısadır,
- Parametre ve değişken değişikliklerine uyarlanabilir ve dinamiktir,
- Çözümlerin doğrulukları daha yüksektir,
- Maliyeti düşüktür ve elde edilen sonuçlar daha tutarlıdır,
- Genel olarak analizlerin gerçekleştirilmesi için uzman bilgisi gerektirmez,
- Çoklu problemlere uygulanabilir.

Koşullu ifade olarak bilinen bir kuralın genel yapısı "EĞER koşul(lar) O HALDE sonuç" yapısındaki bir önermedir. Bu önermede, kuralın önde gelen (koşullar) kısmı tahminleyici değişkenlerin mantıksal kombinasyonlarını içerir ve kuralın sonra gelen (sınıf/sonuç) kısmı, kuralın önde gelen kısmını sağlayan durumlar için tahmin edilen sınıfı içerir. Bu şekilde bir gösterim, keşfedilen bilginin anlaşılabilirliğine katkıda bulunan yüksek düzeyde ve sembolik bilgi gösterim avantajı sağlar.

Sınıflandırma kuralları tanımlama (characterization) kuralları ve ayırt etme (discrimination) kuralları olmak üzere ikiye ayrılır [67]. Tanımlama kurallarında amaç bir kavramın özelliklerini tanımlayan kuralları bulmaktadır ve elde edilen kurallar "EĞER kavram O HALDE özellik" formundadır. Ayırt etme kurallarında ise amaç bir kavrama (sınıfa) ait örneklerin (veri kayıtlarının) kalan örnekler (veri kayıtları veya sınıfları) içinden seçilmesini (ayırt edilmesini) sağlayan kuralların bulunmasıdır. Ayırt etme kuralları "EĞER özellik O HALDE kavram" formunda bulunurlar. Tanımlama kurallarının tersi ayırt etme kuralları değildir.

Çıkarılan kuralların kullanışlı olması için, ilgili verinin özelliklerini yeterli doğruluk ile tanımlamaları gerekir. Sınıflandırmada kullanılan veri genellikle her bir örnek için eşit sayıda değişken içeren örnekler kümesiyle ifade edilir. Değişkenler bir örnekçe temsil edilen bir kavramı tanımlayan farklı özelliklerdir. Bir özelliğin kavram niteliği değişkenler kümesi ve bunların verilen bir kavram için belirleyici olan ilgili değerleri ile tanımlanır. Sınıf (karar) değişkeni olarak ifade edilen özel bir d_c değişkeni içeren, n kayıttan oluşan bir veri kümesi ele alındığında, d_c değişkeni veri kümesindeki kayıtları, sınıflar olarak ifade edilen ve kayıtları sınıflandıran, kayıtları sınıf değişkeninin

değerine göre tanımlanan ayrık alt kümelere ayıran parçalara böler. Burada ayrık alt küme sayısı, veri kümesinde mevcut sınıf sayısına eşittir.

2.11.2. Sınıflandırma Kural Çıkarımı

Sınıflandırma kural çıkarımı, doğru bir sınıflandırıcı oluşturmak için veritabanında küçük bir kurallar kümesi keşfetmeyi amaçlar [41, 68]. Kural çıkarımı süreci, bir veri kümesinden sembolik, sürekli veya kesikli bilginin, veri kümesinde saklı olan bilgiyi yeterli doğrulukta tanımlayan sembolik önermeli mantık kurallarına çevrilmesi olarak düşünülebilir [69, 70].

Kural çıkarımında temel amaç, verideki gizli bilgiyi ortaya çıkarıp anlaşılır şekilde ifade etmek, daha önce bilinmeyen ilişkileri ortaya çıkarmak, nedenleme ve tanımlama kabiliyeti sağlamaktır [2]. Sınıflandırma kural çıkarımı ile elde edilen kurallar şu özelliklere sahip olmalıdır [67];

- Anlaşılır olmalı,
- Kısa, basit, açık ve temiz olmalı
- Çıkarıldığı veriyi doğru tanımlamalı,
- Kurallar tekrarlanmamalı,
- İfade ettikleri bilgi kullanışlı olmalı,
- Verideki bilgiyi özetlemeli.

Veritabanlarından kural çıkarımı, sadece kural tabanlı sınıflandırma için hizmet etmez aynı zamanda keşfedilen dilsel bilgi ile ele alınan probleme bakış açısı sağlar ve daha iyi anlaşılmasına olanak tanır. Kural tabanlı sınıflandırma veri madenciliği bakış açısıyla, sınıflandırma kararının anlaşılabilirliği ile birçok klasik sınıflandırma tekniğine üstünlük sağlamıştır [71].

Bir veri kümesinde sınıflara ait örnekleri sınıflandırma işlemini gerçekleştiren bir kural çoğu zaman "sınıflandırıcı" olarak adlandırılır. Sınıflandırma kural çıkarım teknikleri kural tabanlı metotlar ve kural tabanlı olmayan metotlar olarak ikiye ayrılır [3].

Kural tabanlı metotlar: Kural tabanlı sınıflandırma metotları veriden gizli bilgiyi doğrudan çıkarırlar ve kullanıcılar bu bilgileri kolaylıkla anlayabilirler. C4.5, karar tabloları vb. kural tabanlı metotlara örnek olarak verilebilir.

Kural tabanlı olmayan metotlar: Kural tabanlı olmayan sınıflandırma metotları genellikle kural tabanlı sınıflandırma metotlarına göre daha doğru sonuçlar verirler fakat kara kutu gibi davrandıklarından dolayı elde ettikleri bilgileri kullanıcılar anlayacağı biçimde sunamazlar. Genel olarak yapay sinir ağları gibi kural tabanlı olmayan sınıflandırıcılar çok iyi sınıflandırma doğrulukları elde edebilmelerine rağmen anlaşılabilirlik yönünden rekabetçi değillerdir. Destek vektör makineleri, yapay sinir ağları, doğrusal genetik programlama kural tabanlı olmayan metotlara örnek olarak verilebilir

Bir veri kümesini sınıflandırmak için kural kümesi oluşturulurken, yerel, küresel veya yerel-küresel bir melez algoritma eğitim ve test etme olmak üzere iki aşamada kullanılır. Birinci aşamada sınıflandırılmış kayıtlardan oluşan bir veri kümesinden sınıflandırma kuralları öğrenilir. Bu adımda kullanılan veri kümesine "eğitim veri kümesi" denir. Bu tip öğrenmede öğrenilene ise "kavram" denir. Dolayısıyla bu süreç "kavram öğrenme" olarak adlandırılır [72]. İkinci aşamada, birinci aşamada öğrenilen kurallar bir test veri kümesine uygulanır ve kuralların doğruluğu değerlendirilir. Eğitim veri kümesi gibi, test veri kümesi de sınıf değerlerini içerir. Test kümesi sınıflandırıcının genelleme yeteneğini ve tahminleyici doğruluğunu değerlendirmek için kullanılır. Tahminleyici doğruluk test veri kümesindeki doğru sınıflandırılan örneklerin yüzdesini ifade eder. Eğitim ve test veri kümelerinin seçim süreci oldukça karmaşık ve zor bir süreçtir [67].

2.11.3. Sınıflandırma Metotlarının Değerlendirilmesinde Temel Ölçütler

Veri madenciliğinde belki de en yaygın kullanıma sahip teknik sınıflandırmadır [73]. Sınıflandırmada kategorik bir hedef değişken vardır. Veri madenciliği modeli bu hedef değişkenle ilgili bilgileri ve aynı zamanda girdi değişkenlerini de içeren çok sayıda kayıtlar kümesini ele alır. Sınıflandırmada amaç yeni kayıtlar için hedef değişkenle ilgili bir sınıflandırma çıkarmaktır. Eğitim veri kümesi kullanılarak algoritma hedef değişken ile ilgili bilgileri öğrenir, daha sonra ise test kümesinde bulunan yeni kayıtlara göre öğrendiği bilgileri test eder. Ancak algoritmanın işleyişi ve sonuçların kalitesi hakkında bazı değerlendirmelerin yapılması gerekir. Aşağıda sıralanan ölçütler sınıflandırma metotlarının değerlendirilmesinde yaygın olarak kullanılmaktadır [72].

- Tahminleyici doğruluk,

- Kural anlaşılabilirliği,
- Hız ve ölçeklenirlik,
    - Modeli kurmak için gerekli süre,
    - Modeli kullanmak için gerekli süre.
- Sağlamlık,
    - Gürültüyü ve boş değerleri ele alma
- İlgi çekicilik.

***Tahminleyici doğruluk*:** Tahminleyici doğruluğun belirlenmesi, sınıflandırıcının daha önce görülmemiş örnekleri hangi doğrulukta keşfedeceğini belirleyeceğinden dolayı çok önemlidir. Ayrıca tahminleyici doğruluk farklı sınıflandırıcıların karşılaştırılmasında da kullanıcıya büyük kolaylık sağlar. "Direnme (holdout)", "çapraz-doğrulama (cross-validation)", "önyükleme (bootstrapping)" ve "bir-dışarda-bırak (leave-one-out)" doğruluk tahmininde kullanılan tekniklerdir [1].

Direnme ve çapraz-doğrulama, sınıflandırıcı doğruluğunu değerlendirmede en yaygın kullanıma sahip yöntemlerdir. Bu tekniklerde ele alınan veri rastgele parçalara ayrılır. Direnme metodunda, veri eğitim ve test kümeleri olmak üzere birbirinden bağımsız iki parçaya ayrılır. Genellikle verinin üçte ikisi eğitim kümesi, kalan üçte biri ise test kümesi olarak değerlendirilir. Eğitim kümesi sınıflandırıcı elde etmek için kullanılır ve sınıflandırıcının doğruluğu test kümesi ile elde edilir. Başlangıç verisinin sadece bir kısmı sınıflandırıcı elde etmek için kullanıldığından dolayı direnme tekniği kötümser bir tekniktir. Rastgele alt-örnekleme, direnme metodunun k kez tekrarlandığı, direnme metodunun bir varyasyonudur. Doğruluk, her bir iterasyondan elde edilen doğrulukların ortalamalarının alınması ile hesaplanır.

k-katlı çapraz-doğrulama tekniğinde veri eşit büyüklükte rastgele k parçaya ayrılır. Eğitim ve test işlemleri k kez tekrarlanır. Her bir tekrarda k parçadan farklı bir tanesi test kümesi, kalan k-1 parça ise eğitim kümesi olarak kullanılır. Doğruluk tahmini k iterasyondan elde edilen doğru sınıflandırmaların toplam sayısının başlangıç verisi örnek sayısına bölünmesi ile hesaplanır. Genel olarak 10-katlı çapraz-doğrulama, düşük eşik ve varyansından dolayı sınıflandırıcı doğruluğunun belirlenmesinde önerilen bir tekniktir. Çapraz doğrulama tekniğinde kat olarak en çok 10 kullanılmasının nedeni, farklı öğrenme teknikleri ile çok sayıda veri kümesi üzerinde yapılan detaylı testlerde 10

sayısının en az hatayı veren, doğru sayı olarak çıkmasıdır. Ayrıca bunu destekleyecek teorik bilgiler de mevcuttur [53].

Önyükleme metodu ise değiştirmeli örneklemeye dayanan istatistiksel bir yöntemdir. Daha önce anlatılan yöntemlerde eğitim ve test veri kümeleri değiştirme olmaksızın oluşturulurken yani aynı örnek bir kere seçildikten sonra tekrar seçilemezken önyüklemedeki temel fikir eğitim kümesini oluşturmak için değiştirme ile veri kümesinin örneklenmesidir.

Bir-dışarda-bırak yöntemi, n veri kümesindeki örnek sayısını göstermek üzere, basitçe n-katlı-çapraz-doğrulamayı ifade eder. Bu yöntemde sırasıyla her bir örnek dışarıda bırakılır ve öğrenme metodu kalan örneklerle eğitilir. Test işlemi dışarıda bırakılan örnek üzerinde yapılır, doğruluk değeri ya pozitif ya negatif çıkacaktır. Her bir örnek üzerindeki testin sonucu, yani n testin sonucu ortalanır ve bu ortalama değer son doğruluk değerini ifade eder.

***Kural anlaşılabilirliği*:** Sınıflandırma kural çıkarımında önemli bir diğer değerlendirme kriteri modelin anlaşılabilirliğidir. Modelin anlaşılabilir olması, kullanıcılar tarafından kolayca anlaşılıp yorumlanabilmesi anlamına gelmektedir. Bazı uygulamalarda doğruluk oranlarındaki küçük artışlar önemli olsa da, kuralın yorumlanabilmesi çok daha büyük önem taşıyabilir.

***Hız ve ölçeklenirlik*:** Hız ve ölçeklenirlik sınıflandırma kural çıkarımında elde edilen kuralların değerlendirilmesinde yararlanılan ölçütlerdir. Bu ölçütler modeli kurmak için gerekli olan süre ve modeli kullanmak için gerekli olan süre ile ilgilidir.

Sağlamlık: Sağlamlık, sınıflandırma metotlarının değerlendirilmesinde kullanılan diğer bir ölçüttür ancak doğruluk ve anlaşılırlık kadar sık kullanılmamaktadır. Sağlamlık, eğitim verisinde veya başlangıç alan bilgisindeki bozukluklara duyarlılığının ölçüsüdür.

***İlgi çekicilik*:** Yukarıda açıklanan ölçütlerin yanı sıra, sınıflandırma kural çıkarımında dikkate alınması gereken bir diğer ölçüt ilgi çekiciliktir. Son kullanıcı için [74];

- Kurallar kullanıcının bilgi ve beklentilerine ters düşüyorsa (beklenmedik kurallar),

- Kullanıcılar kurallarla bir şeyler yapabiliyor ve fayda sağlayabiliyorlarsa (kullanılabilir),
- Kullanıcının daha önceki bilgisine bilgi ekliyorsa (yeni) ilgi çekicidir.

2.11.4. Sınıflandırma Kurallarının Mikro ve Makro Değerlendirilmesi

Kuralların değerlendirilmesi sınıflandırma sürecinde büyük önem taşır. Mevcut kural öğrenme algoritmalarının çoğu tekil kural değerlendirme ölçütüne göredir. Bununla birlikte kural çıkarım sisteminin performansı ve sınıflandırma süreci kurallar kümesinin değerlendirilmesinde dikkate alınır. Bu da tekil kural ve kural kümesi değerlendirme ölçütlerinin birleştirilmesini gerektirir.

Kural değerlendirme ölçütleri 2×2 durumsallık tablosunda, kuralın önde gelen ve sonra gelen kısmı arasındaki ilişki analiz edilerek belirlenir. Tablo 2.1. örnek bir durumsallık tablosunu göstermektedir.

Tablo 2.1. 2×2 Durumsallık tablosu.

	Sınıf	Yanlış sınıf	Toplam
Önde gelen	tp	fp	Önde gelen sağlanan
Yanlış önde gelen	fn	tn	Önde gelen sağlanmayan
Toplam	Sınıf sağlanan	Sınıf sağlanmayan	Veri kümesi

Bir kural, verilen bir eğitim örneğini sınıflandırmakta kullanıldığında; pozitif doğru (tp), pozitif yanlış (fp), negatif doğru (tn) ve negatif yanlış (fn) olmak üzere dört durum ortaya çıkar [3]. Pozitif doğru ve negatif doğru, doğru sınıflandırmaları; pozitif yanlış ve negatif yanlış, yanlış sınıflandırmaları ifade eder.

- Pozitif doğru (*tp*): Kural sınıfın pozitif olduğunu tahmin eder ve verilen örneğin sınıfı da pozitiftir.
- Negatif doğru (*tn*): Kural sınıfın negatif olduğunu tahmin eder ve verilen örneğin sınıfı da negatiftir.
- Pozitif yanlış (*fp*): Kural sınıfın pozitif olduğunu tahmin eder fakat verilen örneğin sınıfı negatiftir.

- Negatif yanlış (*fn*): Kural sınıfın negatif olduğunu tahmin eder fakat verilen örneğin sınıfı pozitiftir.

Yao ve Zhou [75], kural değerlendirme ölçütlerini değerlendirilen kural sayısına göre makro ve mikro değerlendirme olmak üzere ikiye ayırmışlardır. İkinci aşamada ise makro değerlendirme ölçütlerini, bir kümede kurallar arasındaki ilişkiye göre çakışan ve çakışmayan kurallar olarak iki gruba sınıflandırmışlardır. Sınıflandırmaları şu şekildedir;

1. Mikro değerlendirme
    - Karmaşıklık ölçütleri (değişken sayısı gibi)
    - Performans ölçütleri (genellik, güvenilirlik, destek gibi)
2. Makro değerlendirme
    - Çakışmayan kurallar
        - Karmaşıklık ölçütleri (değişken sayısı, kural sayısı gibi)
        - Performans ölçütleri (genellik, güvenilirlik, destek gibi)
    - Çakışan kurallar
        - Tutarlı kurallar
            - Karmaşıklık ölçütleri (değişken sayısı, kural sayısı, çakışma derecesi gibi)
            - Performans ölçütleri (genellik, güvenilirlik, destek gibi)
        - Çelişen kurallar
            - Karmaşıklık ölçütleri (değişken sayısı, kural sayısı, çakışma derecesi gibi)
            - Performans ölçütleri (genellik, güvenilirlik, destek gibi)

Mikro değerlendirme tekil kurallar üzerinedir. Mevcut birçok kural değerlendirme ölçütü mikro değerlendirme için önerilmiştir. Bu ölçütler kural üretiminde durdurma kriterinin belirlenmesinde ve sınıflandırma amacıyla yüksek kaliteli kurallar çıkarılmasında kullanılır. Bununla birlikte, tekil kurallara göre değerlendirme çakışan sonuçların ortaya çıkmasına neden olabilir.

Makro değerlendirme bir kural kümesi üzerinedir. Bir karar vermek için birden çok kural olduğundan dolayı mikro değerlendirmeye göre daha karmaşıktır. Makro değerlendirmede, ele alınan uzayda eğer bir nesne kural kümesinde en fazla bir kuralı

sağlıyorsa kurallar çakışmayan kurallardır, eğer birden fazla kuralı sağlıyorsa çakışan kurallardır. Çakışan kuralları da tutarlı ve çelişen kurallar olmak üzere ikiye ayırmışlardır. Ele alınan uzayda eğer bir nesne bir veya daha fazla kuralı aynı sınıfla sağlıyorsa kurallar tutarlıdır, eğer birden çok kuralla en az iki farklı sınıfı sağlıyorsa çelişen kurallardır.

Sınıflandırmalarındaki son aşama farklı kural ölçütlerinin amacına göre değerlendirmedir. Eğer ölçüt kuralların karmaşıklığına göre tasarlanmışsa buna karmaşıklık ölçütü, kuralların performansına göre tasarlanmışsa buna da performans ölçütü demişlerdir. Tekil kuralların karmaşıklık ve performans ölçütleri eğer kurallar çakışmayan ve tutarlı kurallarsa doğrudan kural kümesine uygulanabilir. Çelişen ilişkilerde ise mikro ve makro değerlendirmeleri yapmak gerekir.

Mikro değerlendirme ölçütleri tekil kuralların gücünü göstermek için tasarlanmıştır. En çok kullanılan mikro performans değerlendirme ölçütlerine ilişkin formüller (2.2.)-(2.5.)'de verilmiştir.

$$Doğruluk = \frac{tp+tn}{tp+tn+fp+fn} \tag{2.2.}$$

Doğruluk ele alınan kuralın veri kümesini ne derecede yansıttığını ölçen bir ölçüttür ve veri kümesindeki toplam örnek sayısının doğru sınıflandırılan örnek sayısına bölünmesi ile bulunur.

$$Güvenilirlik = \frac{tp}{tp+fp} \tag{2.3.}$$

Sınıflandırmada güvenilirlik, kuralın önde gelen kısmını sağlayan nesneler içinde hem sınıfı hem de önde gelen kısmı sağlayan nesnelerin oranını ifade eder ve 0 ile 1 arasında değerler alır.

$$Destek = \frac{tp}{tp+fn} \tag{2.4.}$$

Destek ölçütü bir kuralın kabul edilebilirliğinin ölçüsüdür ve 0 ile 1 arasında değerler alır. Sınıflandırma problemlerinde destek sınıfı doğru sağlayan nesneler içinde önde gelen kısmı da sağlayan nesnelerin oranını verir.

$$Genellik = \frac{tp+fp}{N} \quad (2.5.)$$

Genellik, tüm veri kümesinde kuralın önde gelen kısmını sağlayan parçayı ifade eder. Güvenilirlik ve destek gibi genellik değerlendirme ölçütü de (0-1) arasında değerler alır. Mikro kurallarda karmaşıklık ölçütü olarak değişken sayısı kullanılabilir. Bu ölçüt ise kuralın önde gelen kısmında bulunan değişken sayısı ile belirlenir.

Makro değerlendirme, sistemin her bir tekil kuralının performansını değerlendirmek yerine tüm kural çıkarım sisteminin performansını değerlendirmeye odaklanır. Makro performans değerlendirme ölçütleri şu şekilde hesaplanabilir. Doğruluk;

$$Doğruluk = \frac{Kural\ kümesi\ tarafından\ doğru\ sınıflandırılan\ örnek\ sayısı}{Veri\ kümesindeki\ örnek\ sayısı} \quad (2.6.)$$

Makro değerlendirmede doğruluk kural kümesinin doğruluğunu ifade eder ve veri kümesindeki toplam örnek sayısının doğru sınıflandırılan örnek sayısına bölünmesi ile bulunur. Güvenilirlik;

$$Güvenilirlik = \frac{Kural\ kümesi\ tarafından\ doğru\ sınıflandırılan\ örnek\ sayısı}{Kural\ kümesi\ tarafından\ sınıflandırılan\ örnek\ sayıs} \quad (2.7.)$$

Güvenilirlik kural kümesi tarafından sınıflandırılan nesneler içinde doğru sınıflandırılan nesnelerin oranını verir. Destek ise tüm nesneler içinde kural kümesi tarafından doğru sınıflandırılan nesnelerin oranını verir (2.8.);

$$Destek = \frac{Kural\ kümesi\ tarafından\ doğru\ sınıflandırılan\ örnek\ sayısı}{Veri\ kümesindeki\ örnek\ sayısı} \quad (2.8.)$$

Bir kural kümesinin genelliği, veri kümesindeki nesneler içinde kural kümesi tarafından kapsanan nesnelerin oranını gösterir ve aşağıdaki formül ile hesaplanır;

$$Genellik = \frac{Kural\ kümesi\ tarafından\ kapsanan\ örnek\ sayısı}{Veri\ kümesindeki\ örnek\ sayısı} \quad (2.9.)$$

Makro değerlendirmede karmaşıklık ölçütü olarak değişken sayısı kural kümesindeki değişken sayısı hesaplanarak bulunur. Kural sayısı ise kural kümesindeki kural sayısı ile ifade edilir.

2.11.5. Kural Gösterimi

Sınıflandırma kural çıkarımında ilk verilmesi gereken karar bir çözümde kaç kuralın kodlanacağıdır [76]. Bu problemle ilgili iki yaklaşım söz konusudur: *Michigan* yaklaşımı ve *Pittsburgh* yaklaşımı. Michigan yaklaşımında her bir çözüm dizisi bir kural içerirken Pittsburgh yaklaşımda birçok kural bir diziyi meydana getirir.

Pittsburgh yaklaşımı, bütün kural kümesini değerlendirme yeteneğinden dolayı sınıflandırmaya daha uygundur ve bunun sonucu olarak kural etkileşimlerini dikkate alır. Bu yaklaşımla elde edilen kuralların kalitesinin değerlendirilmesi kolaydır. Bununla birlikte, karmaşıktır ve çözümlerin dizi uzunluğunun fazla olmasından dolayı işlem zamanı yüksektir.

Michigan yaklaşımı, karmaşık olmayan çözümler içerdiğinden dolayı düşük işlem zamanı avantajına sahiptir. Bununla birlikte, bir anda bir kuralı ele almasından dolayı kural etkileşimlerini dikkate almaz ve bu da kural kümesinin tahminleyici doğruluğunun temel engelidir. Bu yaklaşımla ilgili diğer bir olumsuzluk ise bir kural kümesi gerekmesine rağmen, yaklaşımın tek bir çözüme yakınsamasıdır. Bu durumla ilgilenmenin bir yolu, farklı kurallar elde etmek için algoritmayı birçok kez çalıştırmaktır. Ancak bu da zaman ve işlem maliyetini artırır.

Çözüm dizisinde tek bir kural mı yoksa çoklu kurallar mı kodlanacağına karar verildikten sonra, dizilerin nasıl kodlanacağına karar verilmelidir. Yüksek düzeyli kural gösterimi kullanılabileceği gibi, ikili kodlama en çok kullanılan ve en basit kodlama şeklidir [72]. Sembolik değişkenler (kesikli değerler içeren değişkenler) için bir yaklaşım her bir değere bir bit kullanmaktır. Bu durumda N değer içeren bir değişken N bitlik bir alt dizi ile gösterilecektir. Örnek olarak, {kırmızı, beyaz, siyah, sarı} değerlerini içeren bir renk değişkeni ele alınırsa “1010” alt dizisi rengin kırmızı veya siyah olduğunu temsil edecektir. Bu yaklaşımda, bir değişkenin eğer tüm alt dizilerinin bit değerlerleri 1 ise, bu değişkenin test neticesi sürekli doğru dönecektir, başka bir ifade ile bu değişken kuralın önde gelen kısmının bir parçası olmayacaktır. Bu kuralın genelliği dikkate alındığında kritik bir faktördür. Bazı değişkenlerin çıkarılması daha basit kurallar ortaya çıkarır ve çakışmaları engeller.

İkili kodlama için diğer bir yaklaşım ise bir değişkenin indeks değerini ikili formda göstermektir. Bu gösterimle, özellikle çok sayıda eleman içeren değişkenler için kısa alt diziler elde edilir. Bu yaklaşımda kuraldan değişken çıkarmak için önemsizlik biti gibi farklı bir teknik kullanmak gerekir.

Sürekli (sayısal) değişkenler için genellikle bir kesiklendirme tekniği gerekir. Eğer değişken değerleri tamsayı veya ondalıklı kısımlarda sabit sayı indeksli ise değişkenin ikili değere doğrudan dönüştürülmesi mümkündür. Kullanılabilecek çok basit kesiklendirme teknikleri olmasına rağmen, kesiklendirme işlemi çok karmaşık olabilir ve kuralların başarısını etkileyebilir. Danışmanlı ve danışmansız kesiklendirme olmak üzere iki tür kesiklendirme vardır. Danışmansız kesiklendirmede sınıf bilgisi kullanılmadan değişken değerleri kesiklendirilir, danışmanlı kesiklendirmede ise sınıf değerleri de kesiklendirme işlemine dahil edilerek değişken kesiklendirilir [53].

Kuralın önde gelen kısmı gibi sınıf yani sonuç kısmının gösterimi için de farklı yaklaşımlar kullanılabilir. Tahminlenen sınıf, değişkenler ile aynı şekilde kodlanabilir ve kuralı gösteren ikili diziye ilave edilebilir. Diğer bir yaklaşım algoritma çalışırken sınıfı sabit tutmak ve her bir sınıf için algoritmayı en az bir kez çalıştırmaktır. Bu durumda sınıf kısmının gösterimde bulunması gerekmez. Daha karmaşık bir süreç gerektiren diğer bir teknik, kuralın önde gelen kısmına göre sınıfa karar vermektir. Bunu gerçekleştirmenin bir yolu bireyin uygunluğunu en büyükleyen sınıfı seçmektir. İlk yaklaşım gibi, bu teknik de algoritma çalışırken kuralın sınıf kısmının değişmesine izin verir.

2.11.6. Uygunluk Fonksiyonu

Sınıflandırmada verilmesi gereken ikinci en önemli karar kullanılacak olan uygunluk yani amaç fonksiyonudur. Sınıflandırma kural çıkarımı için uygunluk fonksiyonu çoğu zaman sınıflandırma sürecinin amacına ve kullanılan kural gösterim yapısına göre seçilir. Tahminleyici doğruluğun en büyüklenmesi çoğu zaman ilk amaçtır ve tahminleyici doğruluk basit bir şekilde, doğru sınıflandırılan örneklerin eğitim kümesinde yer alan örneklere oranı olarak hesaplanır. Spears ve De Jong [77], doğru sınıflandırılan örneklere doğrusal olmayan bir eşik sunan aşağıdaki uygunluk fonksiyonunu önermiştir.

$$\text{Uygunluk(dizi i)} = (\text{doğru sınıflandırılan örnekler yüzdesi})^2 \quad (2.10.)$$

Ancak bu fonksiyon tekil kuralların neden olduğu yanlış sınıflandırmaları etkin bir şekilde cezalandıramaz ve bu da performans problemine neden olabilir. Bu, özellikle kurallar Michigan yaklaşımı ile gösterildiğinde doğrudur, çünkü kurallar arasında etkileşimler dikkate alınmaz. Bu problemi azaltmak için, sadece pozitif doğru ve negatif doğruyu (doğru sınıflandırılan örnekler) değil, aynı zamanda pozitif yanlış ve negatif yanlış (yanlış sınıflandırılan örnekler) değerlerinin farklı varyasyonlarını dikkate alan uygunluk fonksiyonları kullanılabilir.

Tahminleyici doğruluk, sınıflandırma kurallarının değerlendirilmesinde tek faktör değildir. Anlaşılabilirlik ve ilgi çekicilik de uygunluk fonksiyonunda kullanılabilir. Anlaşılabilirlik veya basitlik, problem alanı ve kullanıcı tercihlerine göre farklı yollardan ölçülebilir. Kuralın önde gelen kısmında yer alan ifade sayısı en yaygın kullanılan yöntemdir.

İlgi çekiciliğin ölçülmesi genellikle daha karmaşıktır. Mesela Noda vd. [78], kullanıcıların atadığı ağırlıklara bağlı olarak kuralın önde gelen kısmında yer alan her bir değişkenden bilgi kazancı hesaplayan bir yöntem önermiştir.

2.11.7. Sınıflandırma Literatürü

Son yıllarda veri madenciliğinde sınıflandırma ile ilgili çok sayıda çalışma yapılmaktadır. Yapılan bu çalışmaların çoğu sınıflandırma için metasezgisel algoritmaları kullanmaktadır. Metasezgiseller geniş arama uzaylarıyla birçok problemde yüksek kaliteli çözümler elde ederek, etkin modeller kurarlar. Metasezgisellerin bu özelliklerinden ve sınıflandırma kural çıkarımının bir arama problemi, çözümünün değişkenlerin değerleriyle oluşturulan mantıksal durumların bileşenlerinin bir kümesi olması ve sınıf değeri tahmininin geniş bir arama uzayında gerçekleştirilmesinden dolayı [2], araştırmacılar evrimsel algoritmalar ve sürü zekası tekniklerini kural çıkarımına uygulamaktadırlar. Bunların dışında destek vektör makineleri, birliktelik kuralı madenciliği, kaba küme teorisi gibi konular da sınıflandırma literatüründe yer almaktadır.

Evrimsel Algoritmalar: Bir kural çıkarım tekniği olarak evrimsel algoritmaların iki önemli avantajı uzayı geniş ölçüde aramaları ve arama uzayına uyacak probleme özgü

uygunluk fonksiyonu esnekliğidir [72]. Freitas [79], tarafından vurgulanan diğer bir avantajı ise değişken etkileşimlerini ele almaları ile ilgilidir. Sınıflandırma kural çıkarımında evrimsel algoritmaların temel dezavantajı ise uygulama hızlarıdır. Dharl ve arkadaşları [80] tarafından ortaya sürülen diğer iki dezavantajları popülasyona başlangıç değerlerinin verilmesi aynı zamanda arama sürecindeki rastgelelik ve bir kere bulduğu iyi bir çözüme odaklanmaktaki eğilimidir.

Evrimsel algoritmalar sınıflandırma kural çıkarımında birçok araştırmacı tarafından çalışılmış ve pek çok farklı sınıflandırma problemine uygulanmıştır. Freitas [79], veri madenciliği ve bilgi keşfi için özellikle genetik algoritmalar ve genetik programlama olmak üzere evrimsel algoritmalardaki çalışmaları ortaya koyan bir araştırma sunmuştur.

Carvalho ve Freitas [81], veri madenciliğinde küçük parçalı kurallar bulmak için bir karar ağacı/genetik algoritma melezi sunmuşlardır. Büyük parçalara ait örnekleri bir karar ağacı algoritması (C4.5) tarafından üretilen kurallarla sınıflandırmış, küçük parçalara ait örnekleri ise küçük parçalı kuralları keşfetmek için tasarlanmış bir genetik algoritmadan elde edilen kurallar ile sınıflandırmışlardır. Sekiz farklı veri kümesinde algoritmalarını C4.5 algoritmasının üç farklı versiyonu ile karşılaştırmışlar ve daha iyi doğruluk değerleri elde etmişlerdir.

Hsu ve Hsu [82], sınıflandırma geliştirmekte GEC olarak isimlendirdikleri bir evrimsel yaklaşım sunmuşlardır. Algoritmaları basit yapılı genetik algoritmayı kullanmaktadır. Deneysel çalışmaları GEC algoritmasının klasik kural öğrenme algoritmaları ile rekabet edebilecek yetenekte olduğunu göstermektedir.

Tan ve arkadaşları [66], istenmeyen tıbbi durumların daha iyi anlaşılması ve önlenmesi için klinik uygulamalarda kullanılabilecek sınıflandırma kurallarının çıkarımı için iki aşamalı melez evrimsel bir sınıflandırma tekniği önermişlerdir. İlk aşamada melez bir evrimsel algoritma, iyi bir aday kurallar havuzu oluşturarak arama uzayını daraltmaktadır. Daha sonra bu aday kurallar, doğru ve anlaşılır kural kümeleri oluşturmak için ikinci aşamada kural sayısı ve sırası olarak en iyilenmiştir. EvoC ismini verdikleri algoritmalarını iki referans veri kümesi üzerinde analiz etmişlerdir. Elde ettikleri sonuçlar algoritmalarının anlaşılır ve doğru sınıflandırma kuralları üretebildiğini göstermektedir.

Pappa ve Freitas [83], otomatik kural türetme algoritmaları geliştirmek için bir genetik programlama algoritması önermişlerdir. Geliştirdikleri algoritmanın, değişken etkileşimleri ile başa çıkmakta mevcut açgözlü kural türetme algoritmalarından daha iyi olabileceğini öne sürmüşlerdir.

Bojarczuk vd. [76], tıbbi veri kümeleri için sınıflandırma kuralları çıkarmakta yeni bir kısıtlı sözdizimi genetik programlama algoritması geliştirmişlerdir. Geliştirdikleri algoritmayı beş tıbbi veri kümesi üzerinde değerlendirmişlerdir. Deneysel çalışmaları, tahminleyici doğruluk ve anlaşılırlık yönünden algoritmalarının iyi sonuçlar elde ettiğini göstermektedir.

Carvalho ve Freitas [84], sınıflandırma kuralları çıkarmakta bir karar ağacı/genetik algoritma melezi sunmuşlardır. Melez yaklaşımlarında, küçük parçalara ait örnekleri kapsayan kuralları keşfetmek için tasarlanmış iki genetik algoritma geliştirmiş ve büyük parçalara ait örneklerle ilgili kuralları çıkarmakta bir geleneksel karar ağacı algoritması kullanmışlardır. Algoritmalarının performansını 22 gerçek hayat verisinde değerlendirmişlerdir.

Wang ve Zhang [85], sınıflandırma kural çıkarımı için yeni bir bağışıklık algoritması geliştirmişlerdir. Algoritmalarının performansını doğruluk ve basitlik yönünden başka algoritmalarla karşılaştırmış ve daha iyi sonuçlar elde etmişlerdir.

Tan vd. [86], veri madenciliğinde anlaşılır kurallar çıkarmak için bir dağıtılmış çoklu evrimsel sınıflandırıcı (DCC) sunmuşlardır. Algoritmaları farklı parçaları bir arada eşzamanlı olarak ele almakta ve hesapsal iş yükü internetteki çoklu bilgisayarlar arasında paylaşılmaktadır. DCC algoritmasının avantaj ve performansını makine öğrenme deposundan aldıkları çeşitli veri kümeleri üzerinde değerlendirmişlerdir. Elde ettikleri sonuçlar ve karşılaştırmalar, tahminleyici doğruluğun sağlam olduğunu ve mevcut algoritmalar ile rekabet edebildiğini göstermektedir.

Tan ve arkadaşları [3], yaptıkları çalışmada çoklu karar kuralı listeleri çıkarmak için çok amaçlı bir evrimsel algoritma (DOEA) geliştirmişlerdir. Mevcut algoritmalardan farklı olarak DOEA algoritması, farklı sınıflandırma doğruluğu ve kural sayısı içeren baskın olmayan karar listeleri içermek için Pareto baskınlık kavramını kullanmaktadır. Sekiz

farklı veri kümesi üzerinde yaptıkları analizler algoritmalarının rekabetçi bir doğrulukla anlaşılır kurallar çıkarabildiğini göstermektedir.

Tan vd. [71], sınıflandırma kuralları çıkarmakta CORE ismini verdikleri çoklu evrimsel tabanlı bir sınıflandırma tekniği geliştirmişlerdir. Aday kural ve kural kümelerinin sınıflandırma sürecinin farklı aşamalarında ele alındığı mevcut yaklaşımların tersine CORE algoritmasında kurallar ve kural kümeleri eşzamanlı olarak ele alınmaktadır. Algoritmalarını 7 farklı referans veri kümesi üzerinde analiz etmişlerdir ve çoğu veri kümesinde CORE algoritması anlaşılır ve iyi sınıflandırma kuralları çıkarmıştır.

Chen ve Hsu [87], en yüksek tahminleyici doğrulukta bir karar modeli kurmak için tahminleyiciler, ilgili eşitsizlik ve eşik değerlerini eş zamanlı olarak içeren karar kuralları çıkarmakta göğüs kanseri örneklerini belirlemekte genetik algoritmaya dayalı bir yaklaşım sunmuşlardır. Sonuçlarını ticari bir veri madenciliği yazılımı ile karşılaştırmış ve önerdikleri kural çıkarım algoritmasının tahminleyici doğruluk ve modelleme basitliğini geliştirmekte umut vaat eden bir yöntem olduğunu deneysel olarak göstermişlerdir.

Dehuri ve Mall [88], büyük boyutlu veri tabanlarından yüksek doğrulukta ve anlaşılır sınıflandırma kuralları madenlemek için çok amaçlı bir genetik algoritma sunmuşlardır. Geliştirdikleri algoritmayı iyileştirilmiş hücreli pareto genetik algoritma (INPGA) olarak isimlendirmişlerdir. Algoritmalarının kural üretim performansını basit genetik algoritma ve temel hücreli Pareto genetik algoritma ile karşılaştırmışlar ve daha iyi sonuçlar elde etmişlerdir.

Pitangui ve Zaverucha [89], sınıflandırma problemlerinde kesikli veri için yeni bir çaprazlama operatörü geliştirmişlerdir. Ayrıca, nitelik seçiminde yeni bir gösterim mekanizması sunmuşlardır. Sistemleri Michigan ve Pittsburg yaklaşımlarını bir arada kullanmaktadır. Sistemlerinin performansını C4.5 algoritması ile karşılaştırmışlardır. Elde ettikleri sonuçlar algoritmalarının sağlam olduğunu ve basit kurallarla yüksek doğruluk değerleri elde edebileceğini göstermektedir.

Baykasoğlu ve Özbakır [90], kural çıkarımı için yeni bir kromozom gösterimi ve MEPAR-miner olarak isimlendirdikleri çoklu denklem programlamaya dayanan yeni bir çözüm tekniği geliştirmişlerdir. Geliştirdikleri yöntemi 9 farklı ikili ve çok sınıflı

referans veri kümesi üzerinde değerlendirmişlerdir. Geleneksel yöntemlere göre geliştirdikleri algoritma daha iyi sonuçlar elde etmiştir.

Patterson ve Zhang [91], sınıf dengesizliği problemleri içeren ikili sınıflandırma için bir genetik programlama yaklaşımı geliştirmişlerdir. Elde ettikleri sonuçlar, uygunluk fonksiyonu olarak doğruluk kullanıldığında, GP sisteminin baskın sınıfa yöneldiğini göstermiştir. Sınıf dengesizliği probleminin üstesinden gelmek için yeni uygunluk fonksiyonları önermişlerdir. Deneysel çalışmaları önerdikleri her bir uygunluk fonksiyonunun baskın olmayan sınıf performansını artırdığını ve iki sınıf performansının dengelendiğini göstermiştir.

Dehuri vd. [92], büyük boyutlu veritabanlarından sınıflandırma kuralları çıkarmak için bir seçkinci çok amaçlı genetik algoritma (EMOGA) geliştirmişlerdir. Kuralların tahminleyici doğruluğu, anlaşılırlığı ve ilginçliği üzerine yoğunlaşmışlardır. Bu amaçları eş zamanlı olarak en iyilemek için melez bir çaprazlama operatörüyle çok amaçlı bir genetik algoritma geliştirmişlerdir. Sonuçlarını basit genetik algoritma ile karşılaştırmış ve kendi algoritmalarının basit GA üzerindeki üstünlüğünü ortaya koymuşlardır.

Pappa ve Freitas [93], kural çıkarım algoritmalarını otomatik olarak üreten bir çok amaçlı dilsel tabanlı genetik programlama (MOGGP) sistemi sunmuşlardır. Ayrıca sistemleri doğru kural modelleri sunmaktadır. Karşılaştırmalarını bir tek amaçlı genetik programlama ve diğer üç farklı kural çıkarım algoritması ile yapmışlardır. Yirmi veri kümesi üzerindeki analizleri sistemlerinin daha iyi sonuçlar verdiğini göstermiştir.

Sürü Zekası Teknikleri: Sınıflandırma kural çıkarımında kullanılan diğer bir alan sürü zekası teknikleridir. Karınca koloni optimizasyonu algoritması, parça sürü optimizasyonu algoritması etkin arama kabiliyetlerinden dolayı sınıflandırmada kaliteli sonuçlar üretmektedirler.

Parpinelli vd. [2,11], veri madenciliği için Ant-Miner ismini verdikleri yeni bir algoritma geliştirmişlerdir. Algoritmalarının amacı veriden sınıflandırma kuralları çıkarmaktır. Ant-Miner gerçek karınca sürülerinin davranışlarından ve bazı veri madenciliği kavram ve prensiplerinden esinlenilerek geliştirilmiştir. Algoritmalarını

CN2 algoritması ile altı yerel alan veri kümesinde karşılaştırmışlardır. Doğruluk ve basitlik yönünden daha iyi sonuçlar elde etmişlerdir.

Liu ve arkadaşları [94], Parpinelli vd. [2] tarafından geliştirilen Ant-Miner algoritmasının feremon güncelleme fonksiyonunda iyileştirmeler yaparak, geliştirdikleri algoritmaya Ant-Miner3 ismini vermişlerdir. Algoritmalarını iki veri kümesi üzerinde test etmiş ve orijinal Ant-Miner algoritmasından daha iyi sonuçlar verdiğini görmüşlerdir.

Sousa vd. [95], veri madenciliğinde parça sürü optimizasyonunun kullanımını ele almışlardır. Araştırmalarının ilk aşamasında üç farklı parça sürü veri madenciliği algoritmasını genetik algoritma ve J48 algoritması ile karşılaştırmışlardır. Elde ettikleri sonuçlar neticesinde PSO'nun veri madenciliğinde etkin kullanılabileceğini vurgulamış ve ikinci aşamada PSO algoritmalarından birini iyileştirmişlerdir. Sonuç olarak PSO'nun iyi bir VM aracı olduğunu ileri sürmüşlerdir.

Wang ve Feng [96], sınıflandırma kuralı madenciliği için ACO-Miner ismini verdikleri bir iyileştirilmiş karınca koloni optimizasyonu algoritması geliştirmişlerdir. Algoritmalarının amacı Ant-Miner'a göre daha doğru ve basit kurallar elde etmektir. ACO-Miner algoritmasının performansını Ant-Miner algoritması ile karşılaştırmışlar ve tahminleyici doğruluk ve basitlik açısından daha iyi sonuçlar elde etmişlerdir.

Holden ve Freitas [97], sınıflandırma kurallarının keşfi için yeni bir parça sürü optimizasyonu/karınca koloni optimizasyonu melezi PSO/ACO algoritmasını sunmuşlardır. Geleneksel PSO algoritmasından farklı olarak, geliştirdikleri melez algoritma nominal değerleri ön işleyerek sayılara dönüştürme ihtiyacı duymadan nominal değişkenlerle başa çıkabilmektedir.

Smaldon ve Freitas [98], Unordered Rule Set Ant-Miner olarak isimlendirdikleri sırasız sınıflandırma kuralları üreten algoritmayı orijinal Ant-Miner algoritmasının bir versiyonu olarak geliştirmişlerdir. Algoritmalarını 6 veri kümesi üzerinde orijinal Ant-Miner algoritması ile karşılaştırmışlardır. Tahminleyici doğruluk yönünden karşılaştırılabilir sonuçlar elde etmişlerdir.

Admane vd. [99], karıncaları kullanarak danışmansız sınıflandırma problemleri için AntPart ismini verdikleri bir algoritma geliştirmişlerdir. Algoritmalarının performansını yine karınca davranışlarından esinlenilerek geliştirilen üç farklı karınca algoritması ile karşılaştırmışlardır.

Ji ve arkadaşları [100], temel Ant-Miner algoritması üzerinde bazı iyileştirmeler yapmışlardır. İlk olarak kural ve durum sayılarını azaltmak için bir kural cezalandırma operatörü uygulamışlardır; ikinci olarak ise yakınsama oranını hızlandırmak için bir mutasyon operatörü ve bir adaptif durum çevrim kuralı uygulamışlardır. Deneysel çalışmaları algoritmalarının daha uygun ve kaliteli kuralları temel Ant-Miner algoritmasından daha kısa sürede bulduğunu göstermektedir.

De Falco vd. [101], çok sınıflı veritabanlarında sınıflandırma problemini çözmekte parça sürü optimizasyonu algoritmasının kullanımını ele almışlardır. PSO'nun üç farklı versiyonuyla üç farklı uygunluk fonksiyonunu değerlendirmişlerdir ve en iyi için PSO'yu dokuz farklı teknik ile karşılaştırmışlardır. Elde ettikleri sonuçlar PSO algoritmasının rekabetçi olduğunu göstermiştir.

Martens vd. [102], sınıflandırma için AntMiner+ ismini verdikleri bir algoritma geliştirmişlerdir. Daha önce geliştirilen AntMiner algoritmalarından farklı olarak algoritmaları, MAX-MIN karınca sistemini daha iyi kullanmaktadır. Deneysel çalışmaları ve karşılaştırma sonuçları, geliştirdikleri algoritmanın kaliteli sonuçlar üretebildiğini göstermektedir.

Zahiri ve Seyedin [103], etkin bir sürü zekası tabanlı sınıflandırıcı geliştirmekte parça sürü optimizasyonu algoritmasından faydalanmışlar ve geliştirdikleri algoritmayı IPS-classifier olarak adlandırmışlardır. Üç farklı veri kümesi üzerindeki analizleri algoritmalarının çok katmanlı algılayıcılardan ve k-en yakın komşu algoritmasından daha iyi sonuçlar elde ettiğini göstermektedir.

Holden ve Freitas [104], daha önce geliştirmiş oldukları orijinal PSO/ACO algoritması üzerinde iyileştirmeler yapmışlardır. Yeni algoritmalarını PSO/ACO2 olarak isimlendirmişlerdir. Deneysel çalışmaları geliştirdikleri algoritmanın rekabetçi bir sınıflandırıcı olduğunu göstermektedir.

Jaganathan vd. [105], veri ön işleme için iyileştirilmiş hızlı azaltım algoritması ve Ant-Miner algoritmasını birleştiren bir sistem önermişlerdir. Standart veri kümeleri üzerinde geliştirdikleri algoritmayı test etmiş ve orijinal Ant-Miner algoritmasından daha iyi sonuçlar elde etmişlerdir.

Thangavel ve Jaganathan [106], sınıflandırma kuralı çıkarımına TACO-miner ismini verdikleri yeni bir karınca koloni optimizasyonu algoritması önermişlerdir. Algoritmalarının temel amacı, basit ve yüksek doğrulukta kurallar içeren anlaşılır sınıflandırma kuralları çıkarmaktır. Geliştirdikleri algoritmayı, literatürde yer alan ve sınıflandırmada karınca koloni optimizasyonu algoritmasını kullanan algoritmalarla karşılaştırmışlardır. Doğruluk, basitlik, işlem zamanı ve maliyeti açısından daha iyi sonuçlar elde etmişlerdir.

Otero ve arkadaşları [107], sürekli değişkenlerle başa çıkmakta Ant-Miner algoritmasının bir versiyonu olan cAnt-Miner algoritmasını geliştirmişlerdir. Sürekli değişkenleri ele almak için kural kurulum sürecinde entropi tabanlı kesiklendirme metodunu kullanmışlardır. Algoritmalarını Ant-Miner algoritması ile 8 referans veri kümesi üzerinde tahminleyici doğruluk ve basitlik ölçütleri açısından analiz etmişlerdir. Elde ettikleri sonuçlara göre geliştirdikleri algoritma daha doğru ve basit sınıflandırma kuralları üretmektedir.

Nalini ve Balasubramanie [108], Ant-Miner algoritmasına iki katkı ile algoritmayı geliştirmişlerdir. İlk olarak Laplas-doğrulanmış güvenilirlik sezgisel fonksiyonunu karma değişkenler için sıralanmamış kural kümeleri üretmekte ve keşfedilmiş bir kuralın tahminleyici doğruluğunu iyileştirmekte mevcut kısmi kuralda bir ifade eklemek için deterministik seçim ve stokastik seçimi kullanmışlardır. İkinci olarak ise, kesiklendirme kullanmaksızın sürekli ve kesikli değişkenleri ele almışlardır. Algoritmalarının performansını orijinal Ant-Miner algoritması ile karşılaştırmışlar ve daha kısa sürede daha doğru ve kısa kurallar elde etmişlerdir.

Otero vd. [109], daha önce geliştirmiş oldukları cAnt-Miner algoritmasına iki yeni metot daha sunmuşlardır. İlk metot sürekli değişken aralıklarının gösteriminde daha esnektir. İkinci metot ise problem değişken etkileşimlerini araştırmaktadır. Sekiz veri kümesi üzerindeki deneysel değerlendirmeleri sundukları algoritmaların daha doğru sınıflandırma modelleri keşfettiğini göstermektedir.

Birliktelik Kuralları: Birliktelik kuralları kullanarak sınıflandırma, birliktelik kuralları ve sınıflandırmayı birleştirir. Dolayısıyla tek bir sınıf değişkeninin doğru tahminlenmesi ile ilişkilendirilir. Birliktelik kuralı madenciliğinin en önemli avantajı bütün ilginç kuralların bulunabilmesidir. Bununla birlikte veritabanlarında yer alan birlikteliklerin çok sayıda olması genellikle çok fazla kurala neden olur ve bu da doğrudan sınıflandırmada kullanımı etkiler. Sınıflandırmada birliktelik kuralı kullanılırken gerçekleştirilmesi gereken üç adım vardır; doğru kurallar kümesinin madenlenmesi, kuralların değerlendirilmesi ve budanması ve son olarak bulunan kural kümesi ile görülmemiş örneklerin sınıflandırılması.

Birliktelik kuralı madenciliği teknikleri, sınıflandırma kuralı çıkarımında son yıllarda yaygın olarak kullanılmaktadır. Bu çalışmalara birkaç örnek verilecek olursa; Chen vd. [110], sınıflandırma kavramı içinde genişletilmiş birliktelik kuralı madenciliği tekniğine dayanan bir sınıflandırıcı kurmak için yeni bir yaklaşım sunmuşlardır. Algoritmalarını kazanç tabanlı birliktelik kuralı sınıflandırması (GARC) olarak isimlendirmişlerdir. Algoritmalarının performansını 30 referans veri kümesi üzerinde diğer bazı sınıflandırıcılar ile karşılaştırmış ve daha kaliteli çözümler elde etmişlerdir.

Thabtah ve Cowling [111], çoklu etiketli kurallar çıkaran, sıralanmış çoklu etiketli kural (RMR) ismini verdikleri yeni bir birliktelikçi sınıflandırma tekniği geliştirmişlerdir. Algoritmaları kuralların çakışmasını engellemektedir. Yirmi farklı veri kümesi üzerindeki deneysel çalışmaları sistemlerinin, çoklu sınıflarla ilgili kurallar içeren sınıflandırıcılar üretebildiğini göstermektedir.

Coenen ve Leng [112], birliktelik kuralı madenciliği parametrelerinin, sınıflandırma tahminleyici doğruluğu üzerindeki etkilerini incelemişlerdir. Parametrelerin uygun seçilmesiyle doğruluğun iyileşeceğini göstermiş ve en iyi parametre seçimi için bir tırmanış metodu sunmuşlardır.

2.12. Sonuç

Bu bölümde, tez çalışmasının çatısını oluşturan konularla ilgili genel bilgilere yer verilmiştir. Geliştirilen eğitilmiş yapay sinir ağlarından kural çıkarım algoritmasının bir sınıflandırma algoritması ve sınıflandırmanın da veri madenciliği fonksiyonlarından biri olması nedeniyle VM ve sınıflandırma detaylı olarak anlatılmıştır. Sınıflandırmada

kural gösterimi, uygunluk fonksiyonu ve etkinlik ölçütleri, geliştirilen algoritmaya temel oluşturması nedeniyle detaylı olarak açıklanmıştır.

Ayrıca bu bölümde literatürde yer alan evrimsel algoritmalar ve sürü zekası gibi metasezgisellerle kural çıkarım algoritmaları incelenerek özetlenmiştir. Böylece geliştirilen algoritmanın etkinliğinin, deneysel çalışma ve analizler ile ilgili 7. Bölümde bu algoritmalar ile karşılaştırılması mümkün olmuştur.

3. BÖLÜM
YAPAY SİNİR AĞLARI

3.1. Giriş

Bu bölüm, yapay sinir ağlarından kural çıkarımına temel oluşturması için yapay sinir ağlarına ayrılmıştır. İlk olarak YSA'ların tanımı ve tarihçesi verildikten sonra, YSA'ların avantaj ve dezavantajları, uygulama alanlarına değinilmiştir. Biyolojik sinir sistemi kısaca açıklandıktan sonra, biyolojik sinir sistemine benzetilerek geliştirilen yapay sinir hücresi açıklanmıştır.

Ayrıca bu bölümde yapay sinir ağlarının yapısı, aktivasyon fonksiyonları ve öğrenme algoritmaları gibi YSA'ların yapısal özelliklerine değinilmiştir. Son olarak ise yapay sinir ağlarının uygulanmasında karşılaşılan problemlere kısaca yer verilmiştir.

3.2. Yapay Sinir Ağlarının Tanımı ve Tarihçesi

Yapay sinir ağları, insan beyninin öğrenme yolu ile yeni bilgiler üretebilme, oluşturabilme ve keşfedebilme gibi yeteneklerini taklit ederek ve bir ağa bağlı çok sayıda nöron kombinasyonunun matematiksel modelini oluşturarak, sistemlere herhangi bir yardım almadan otomatik olarak öğrenme, genelleme yapma, hatırlama gibi kabiliyetler kazandırmayı amaçlayan bilgi işleme sistemleridir [113, 114]. Bu kabiliyetleri klasik programlama yöntemleri ile gerçekleştirmek oldukça zor veya mümkün değildir. Dolayısıyla, yapay sinir ağlarının, programlanması çok zor veya mümkün olmayan olaylar için geliştirilmiş adaptif bilgi işleme ile ilgilenen bir bilim dalı olduğu söylenebilir [115].

YSA kavramı beynin çalışma prensiplerinin sayısal bilgisayarlar üzerinde taklit edilmesi fikri ile ortaya atılmış ve ilk çalışmalar beyni oluşturan biyolojik hücrelerin ya-

ni nöronların matematiksel olarak modellenmesi üzerine yoğunlaşmıştır. YSA'nın dayandığı ilk hesaplama modelinin temelleri 1943 yılında McCulloch ve Pitts'in çalışmasıyla atılmıştır [116]. Daha sonra 1954 yılında Farley ve Clark tarafından bir ağ içinde uyarılara tepki veren, uyarılara adapte olabilen model oluşturulmuştur. 1960 yılı ise ilk yapay bilgisayarın ortaya çıkış yılıdır. 1963'de basit modellerin ilk eksiklikleri fark edilmiş, ancak başarılı sonuçlar 1970 ve 1980'lerde termodinamikteki teorik yapıların doğrusal olmayan ağlarının geliştirilmesinde elde edilmiştir. 1985 yılı, yapay sinir ağlarının oldukça tanındığı, yoğun araştırmaların başladığı yıl olmuştur [117].

Yapay sinir ağları örnekler yoluyla öğrenmekte ve nöron olarak isimlendirilen, birbirine bağlı işlemci elemanlardan oluşmaktadır. Her bir nöron bağlantısının bir ağırlık değeri vardır ve yapay sinir ağlarının sahip olduğu bilgi, bu ağırlık değerlerinde saklıdır. YSA metodolojisinde, bir YSA mimarisi oluşturulur ve çeşitli matematiksel algoritmalardan bir tanesi kullanılarak üretilen çıktıların doğruluk düzeyinin en büyüklenmesi için gerekli olan ağırlık değerleri belirlenir. YSA'lar mevcut örnekleri kullanarak ağırlıkları belirlemek yoluyla girdi değişkenleri ile çıktı değişkenleri arasındaki ilişkiyi ortaya çıkarırlar. Bu şekilde yapay sinir ağları eğitilir. Bir kez bu ilişkiler ortaya çıkartıldıktan yani ağ eğitildikten sonra, YSA'lar yeni veriler ile çalıştırılabilir ve tahminler üretilebilir. Yapay sinir ağlarının geliştirilmesindeki temel amaç, insanın bilgisayara göre sahip olduğu üstün yönlerin bilgisayarlara aktarılmak istenmesidir.

Genel olarak yapay sinir ağları model seçimi ve sınıflandırılması, kümeleme veya kategorize etme, fonksiyon tahminleme, optimizasyon, gürültülü ve eksik bilgilerin işlenmesi ve veri sınıflandırılması gibi alanlarda başarılıdır [118]. Klasik programlama teknikleri ise özellikle model seçme alanında verimsizdirler ve sadece algoritmaya dayalı hesaplama işlemleri ile kesin aritmetik işlemlerde hızlıdırlar.

3.3. Yapay Sinir Ağlarının Özellikleri

Yapay sinir ağları hesaplama ve bilgi işleme yeteneğini paralel dağıtılmış yapısından, öğrenebilme ve genelleme yapabilme yeteneğinden almaktadır. Genelleme, eğitim veya öğrenme sürecinde karşılaşılmayan girişler için yapay sinir ağının uygun tepkileri üretmesi olarak tanımlanır. Bu üstün özellikleri YSA'ya karmaşık problemleri

çözebilme yeteneği sağlamaktadır. Yapay sinir ağlarının özellikleri şu şekilde sıralanabilir [119];

***Doğrusal Olmama*:** Yapay sinir ağlarının temel işlem elemanı olan hücre doğrusal olmadığı için bu hücrelerin birleşmesinden meydana gelen YSA da doğrusal değildir ve bu doğrusal olmama özelliği tüm ağa yayılmış durumdadır. Bu özelliği ile yapay sinir ağları, doğrusal olmayan karmaşık problemlerin çözümünde önemli bir araçtır.

Öğrenme: Yapay sinir ağları örneklerle çalışır. Yapay sinir ağlarının istenilen davranışı gösterebilmesi için amaca uygun olarak ayarlanması gerekir. Bunun için hücreler arasında doğru bağlantılar kurulmalı ve bağlantılar uygun ağırlıklara sahip olmalıdır. YSA'nın karmaşık yapısından dolayı bağlantılar ve ağırlıklar önceden ayarlanamaz veya tasarlanamaz. Bu nedenle YSA, istenilen davranışı gösterecek şekilde ilgilendiği problemden aldığı eğitim örneklerini kullanarak problemi öğrenmelidir.

***Paralellik*:** YSA, çok sayıda hücrenin çeşitli şekillerde bağlanmasından oluştuğundan paralel dağıtılmış bir yapıya sahiptir ve ağın sahip olduğu bilgi, ağdaki bütün bağlantılar üzerine dağılmış durumdadır. Oysa alışılmış bilgi işlem yöntemlerinin çoğu seri işlemlerden oluşmaktadır. Bu da hız ve güvenirlik sorunlarını beraberinde getirmektedir. Seri bir işlem gerçekleşirken herhangi bir birimin yavaş oluşu tüm sistemi doğrudan yavaşlatırken, paralel bir sistemde yavaş bir birimin etkisi çok azdır.

Genelleme: Yapay sinir ağı örneklerden öğrendikten sonra eğitim sırasında önceden karşılaşmadığı test örnekleri için de istenilen tepkiyi üretebilir.

***Gerçeklenme Kolaylığı*:** Yapay sinir ağlarının basit işlemler gerçekleyen hücrelerden oluşması ve bağlantıların düzgün olması, ağların gerçeklenmesi açısından büyük kolaylık sağlamaktadır.

***Uyarlanabilirlik*:** Yapay sinir ağları ele alınan problemdeki değişikliklere göre ağırlıklarını ayarlar. Yani, belirli bir problemi çözmek amacıyla eğitilen YSA, problemdeki değişiklere göre tekrar eğitilebilir, değişimler devamlı ise gerçek zamanda da eğitime devam edebilir.

***Eksik Bilgi ile Çalışma*:** YSA'lar eğitildikten sonra eksik bilgiler ile çalışabilir ve gelen yeni örneklerde eksik bilgi olmasına rağmen sonuç üretebilirler. Oysa geleneksel sistemler bilgi eksik olunca çalışamazlar.

***Belirsiz, Tam olmayan Bilgileri İşleyebilme*:** Yapay sinir ağlarının belirsiz bilgileri işleyebilme yetenekleri vardır. Olayları öğrendikten sonra belirsizlikler altında ağlar öğrendikleri olaylar ile ilgili ilişkileri kurarak kararlar verebilirler.

***Yerel Bilgi İşleme*:** YSA'da her bir işlem birimi, ele alınan problemin tamamı ile ilgilenmek yerine, sadece problemin gerekli parçası ile ilgilenmektedir ve problemin bir parçasını işlemektedir. Hücreler çok basit işlem yapmalarına rağmen, sağlanan görev paylaşımı sayesinde çok karmaşık problemler çözülebilmektedir.

***Hata Toleransı*:** YSA, paralel dağıtılmış bir yapıya sahiptir ve ağın sahip olduğu bilgi, ağdaki bütün bağlantılar üzerine dağılmış durumdadır. Bu nedenle, eğitilmiş bir yapay sinir ağının bazı bağlantılarının hatta bazı hücrelerinin etkisiz hale gelmesi ağın doğru bilgi üretmesini önemli ölçüde etkilemez. Oysa sayısal bir bilgisayarda, herhangi bir işlem elemanını yerinden almak, onu etkisiz bir makineye dönüştürmektir. Yapay sinir ağlarının hatayı tolere etme yetenekleri geleneksel yöntemlere göre oldukça yüksektir.

***Donanım ve Hız*:** YSA, paralel yapısı nedeniyle büyük ölçekli entegre devre (VLSI) teknolojisi ile gerçeklenebilir. Bu özellik, yapay sinir ağlarının hızlı bilgi işleme yeteneğini artırır ve gerçek zamanlı uygulamalarda arzu edilir.

***Analiz ve Tasarım Kolaylığı*:** Yapay sinir ağlarının temel işlem elemanı olan hücrenin yapısı ve modeli bütün YSA yapılarında hemen hemen aynıdır. Yapay sinir ağlarının farklı uygulama alanlarındaki yapıları da standart yapıdaki bu hücrelerden oluşacaktır. Dolayısıyla, farklı uygulama alanlarında kullanılan YSA'ları, benzer öğrenme algoritmalarını ve teorilerini paylaşabilirler. Bu özellik, farklı problemlerin yapay sinir ağları ile çözülmesinde büyük kolaylık sağlar.

***Sayısal Bilgi ile Çalışma*:** Yapay sinir ağları sadece sayısal bilgiler ile çalışırlar. Sembolik ifadeler ile gösterilen bilgilerin sayısal gösterime çevrilmesi gerekir.

3.4. Yapay Sinir Ağlarının Avantaj ve Dezavantajları

Yapay sinir ağlarının en büyük avantajı, öğrenme kabiliyetinin olması ve farklı öğrenme algoritmaları kullanabilmesidir. Bunun yanı sıra en çok belirtilen dezavantajları ise sistemin çalışmasının analiz edilememesi ve öğrenme işleminde başarısızlık riski olmasıdır [120]. Yapay sinir ağlarının avantajları şunlardır;

- Matematiksel bir modele ihtiyaç duymazlar,
- Kural tabanı kullanımı gerektirmezler,
- Görülmemiş örnekler hakkında bilgi üretebilirler,
- Eksik bilgi ile çalışabilirler,
- Belirsiz, tam olmayan bilgileri işleyebilirler,
- Algılamaya yönelik olaylarda kullanılabilirler,
- Örüntü ilişkilendirme ve sınıflandırma yapabilirler,
- Kendi kendini organize etme ve öğrenebilme yetenekleri vardır,

YSA'ların dezavantajı olarak ise şunlar söylenebilir;

- Sistem içerisinde ne olduğu bilinemez, kara kutu gibidirler,
- Probleme uygun ağ yapısının belirlenmesi genellikle deneme yanılma yoluyla yapılır,
- Bazı ağlarda ağın parametre değerlerinin belirlenmesinde bir kural yoktur,
- Ağın eğitiminin ne zaman bitirileceğine karar vermek için geliştirilmiş bir yöntem yoktur,
- Sadece sayısal bilgiler ile çalışmaktadırlar,
- Bazı ağlar hariç, kararlılık analizi yapılamaz,
- Donanım bağımlı çalışırlar,
- Farklı sistemlere uyarlanması zor olabilir.

3.5. Yapay Sinir Ağlarının Uygulama Alanları

Yapay sinir ağları özellikle çözümü zor ve karmaşık çok farklı problemlerde başarılı bir şekilde uygulanmaktadır. YSA'lar aşağıdaki özellikleri taşıyan alanlarda kullanılmaya uygun bir araçtır;

- Çok değişkenli problem uzayı,

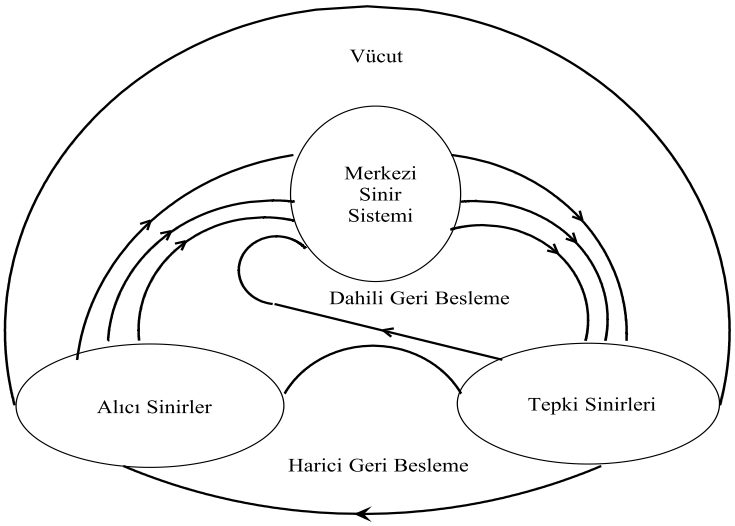
- Probleme ilişkin değişkenler arasında karmaşık etkileşim,
- Çözüm uzayının bulunmaması, tek bir çözümün veya çok sayıda çözümün olması.

Yapay sinir ağları çok farklı alanlara uygulanabilmektedir. Bu alanlara örnek olarak;

- Arıza analizi ve tespiti,
- Tıp,
- Parmak izi tanıma,
- Otomatik araç denetimi,
- Kalite kontrol,
- Savunma sanayi,
- Petrol ve gaz arama,
- İflas tahmini,
- Laboratuar araştırmaları,
- İşlem modelleme ve yönetimi,
- Konuşma ve yapı tanımlama,
- Kredi derecelendirme,
- Ekonomik ve finansal öngörü
- Haberleşme,
- Üretim,
- Otomasyon ve kontrol verilebilir.

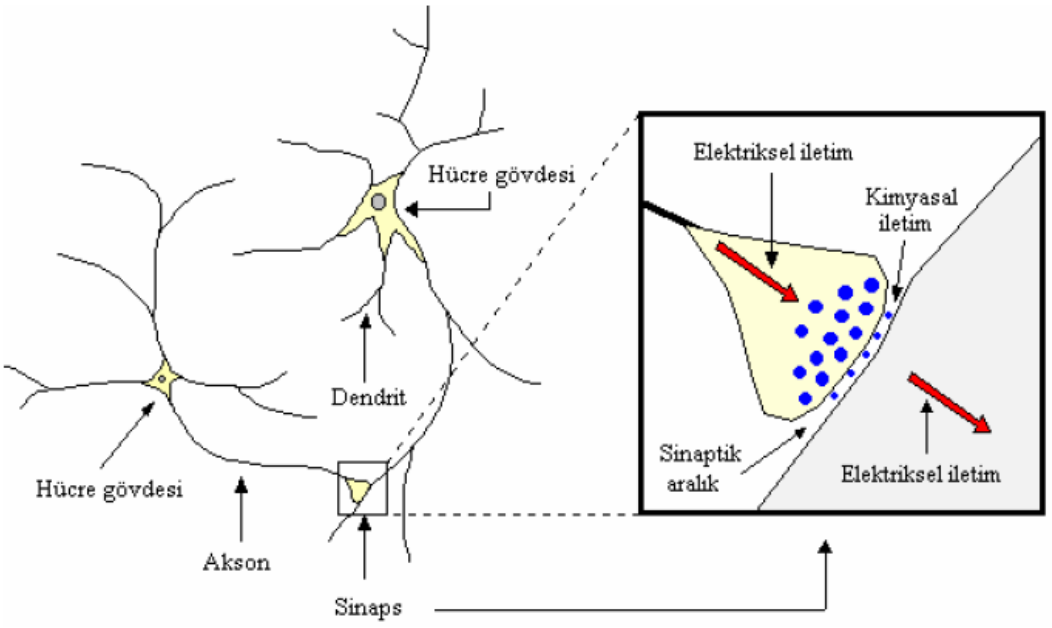
3.6. Biyolojik Sinir Sistemi

Biyolojik sinir sistemi, merkezinde sürekli olarak bilgiyi alan, yorumlayan ve uygun bir karar üreten bir merkez ve bu merkezin kontrolünde bulunan alıcı (resöptör) ve tepki sinirlerinden (efektör) oluşur. Alıcı sinirler, organizma içerisinden veya dış ortamdan aldıkları uyarıları, beyne bilgi ileten elektriksel sinyallere dönüştürler [119]. Tepki sinirleri ise beynin ürettiği elektriksel sinyalleri organizma çıktısı olarak uygun tepkilere dönüştürürler. Merkezi sinir ağında bilgiler, alıcı ve tepki sinirleri arasında ileri ve geri besleme yönünde değerlendirilerek uygun tepkiler üretilir. Bu yönüyle biyolojik sinir sistemi, kapalı çevrim denetim sistemi özelliklerini taşır. Şekil 3.1. bir sinir sisteminin blok diyagramı göstermektedir.



Şekil 3.1. Biyolojik sinir sistemi blok diyagramı.

Merkezi sinir sisteminin temel işlem elemanı nöron olarak adlandırılan sinir hücresidir. Bir nöron, soma (hücre gövdesi), akson, sinaps ve dentrit olmak üzere başlıca dört kısımdan oluşur. Soma yani hücre gövdesi, gelen bilgileri toparlayan, birleştiren ve biçimini değiştirerek diğer sinirlere gönderen yapıdır. Bir nöronda yüzlerce bazen de binlerce bulunabilen dentritin içyapısı sinir gövdesi ile aynıdır. Dentritler kısa lifler olup, diğer hücrelerden aldıkları bilgileri hücre gövdesine bir ağaç yapısı şeklinde ince yollarla iletirler. Her nöronda sadece bir tane bulunan akson ise elektriksel darbeler şeklindeki bilgiyi hücreden dışarıya taşıyan daha uzun liflerdir. Aksonların bitimi, ince yollara ayrılabilir ve bu yollar, diğer hücreler için dentritleri oluşturur. Aksonun dentrite bağlantı elemanı sinapslardır. Sinapsa gelen ve dentritler tarafından alınan bilgiler genellikle elektriksel darbelerdir ancak, sinapsdeki kimyasal ileticilerden etkilenirler. Belirli bir sürede bir hücreye gelen girişlerin değeri, belirli bir eşik değerine ulaştığında hücre bir tepki üretir. Hücrenin tepkisini artırıcı yöndeki girişler uyarıcı, azaltıcı yöndeki girişler ise önleyici girişler olarak bilinir ve bu etkiyi sinaps belirler. Özet olarak basit bir nöron, dentritler yoluyla sinyalleri alan ve bunları somasında işleyen buna karşı gelen cevapları çıkış olarak aksonlar yoluyla ileten basit bir işlem elemanı olarak ifade edilebilir. Şekil 3.2. bir nöron hücresinin yapısını göstermektedir.



Şekil 3.2. Biyolojik nöron.

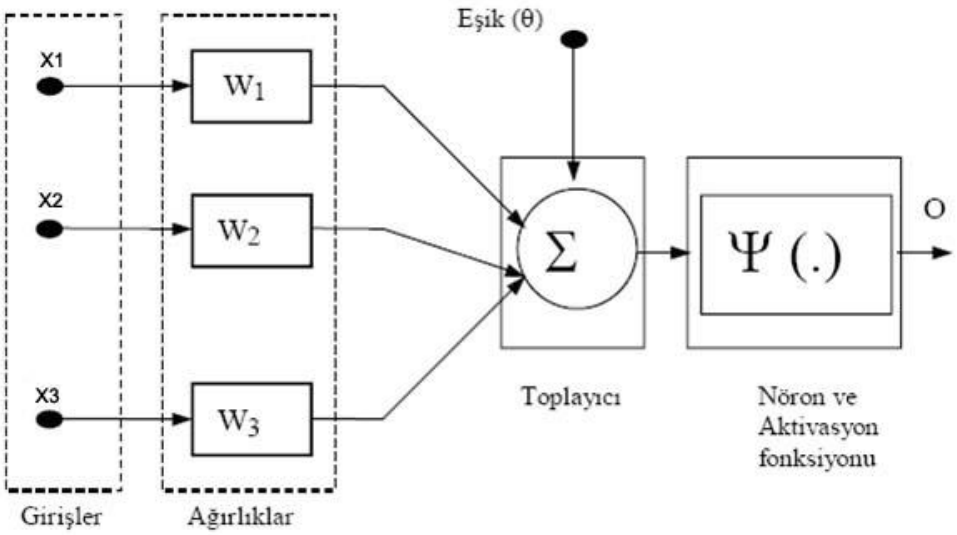
Yapay sinir ağları, insan beyninin çalışma prensibi örnek alınarak geliştirilmiştir. Yapay sinir ağları ile biyolojik sinir sistemi arasında yapısal benzerlikler bulunmaktadır. Bu benzerlikler Tablo 3.1.'de gösterilmektedir.

Tablo 3.1. Sinir sistemi ile YSA'nın benzerlikleri.

Sinir sistemi	YSA sistemi
Nöron	İşlem elemanı
Dendrit	Toplama fonksiyonu
Hücre gövdesi	Aktivasyon fonksiyonu
Aksonlar	Eleman Çıkışı
Sinapslar	Ağırlıklar

3.7. Yapay Sinir Hücresi

Yapay sinir hücresi, YSA'nın çalışmasına esas teşkil eden en küçük bilgi işleme birimidir. Temel bir yapay sinir hücresi biyolojik sinir hücresine göre çok daha basit bir yapıya sahiptir. En temel nöron modeli Şekil 3.3.'de görülmektedir. Yapay sinir hücresi beş temel elemandan oluşmaktadır. Bunlar girdiler, ağırlıklar, toplama fonksiyonu, aktivasyon fonksiyonu ve çıkışlardır.



Şekil 3.3. Temel yapay sinir hücresi.

Girdiler, dış ortamdan ya da diğer nöronlardan alınan bilgilerdir ve ağın öğrenmesi istenen örnekler tarafından belirlenirler. Yapay sinir hücresine dış dünyadan olduğu gibi başka hücrelerden veya kendi kendisinden de bilgiler gelebilir.

Ağırlıklar, dış ortamdan alınan veriyi nörona bağlarlar ve ilgili girdilerin etkisini belirlerler. Ağırlıkların büyük yada küçük olması önemli veya önemsiz olduğu anlamına gelmez. Ağırlığın pozitif veya negatif olması etkisinin pozitif veya negatif olduğunu gösterir. Sıfır olması ise herhangi bir etkinin olmadığını gösterir. Ağırlıklar değişken veya basit değerler olabilirler.

Toplama fonksiyonu, bir hücreye gelen net girdiyi hesaplar. Bunun için değişik fonksiyonlar kullanılmakla birlikte en yagın kullanılan fonksiyon ağırlıklı toplamdır. Burada her gelen girdi değeri kendi ağırlığı ile çarpılarak toplanır, daha sonra ise aktivasyon fonksiyonun eşik değeri bu toplama eklenir. Böylece ağa gelen net girdi bulunur. Bu, *w* ağırlıkları, *x* girdileri, *n* bir hücreye gelen toplam girdi sayısını ve *b* eşik değerini göstermek üzere şu şekilde ifade edilmektedir;

$$NET = \sum_{i=1}^{n} w_i x_i + b \qquad (3.1.)$$

Tablo 3.2. değişik toplama fonksiyonlarıyla ilgili örnekler içermektedir [115]. Tablodan görüldüğü gibi, bazı durumlarda gelen girdilerin değeri dikkate alınırken bazı durumlarda ise gelen girdilerin sayısı önemli olabilmektedir. Bir problem için en uygun

toplama fonksiyonunun belirlemek için bulunmuş bir formül yoktur. Genellikle deneme yanılma yöntemi ile toplama fonksiyonu belirlenmektedir. Bir yapay sinir ağında bulunan işlem elemanlarının tamamının aynı toplama fonksiyonuna sahip olmaları gerekmez. Her işlem elemanı bağımsız olarak farklı bir toplama fonksiyonuna sahip olabileceği gibi hepsi aynı toplama fonksiyonuna sahip olabilir. Bu tamamen tasarımcının kendi öngörüsüne dayanarak verdiği karara bağlıdır.

Tablo 3.2. Toplama fonksiyonu örnekleri.

Net giriş	**Açıklama**
Çarpım $Net\ girdi = \prod_i w_i x_i$	Ağırlık değerleri girdiler ile çarpılır ve daha sonra bulunan değerler birbirleri ile çarpılarak net girdi hesaplanır.
Maksimum $Net\ girdi = Maks(w_i x_i),$ i=1…..N	N adet girdi içinden ağırlıklar ile çarpıldıktan sonra en büyüğü yapay sinir hücresinin net girdisi olarak kabul edilir.
Minimum $Net\ girdi = Mak(w_i x_i),$ i=1…..N	N adet girdi içinden ağırlıklar ile çarpıldıktan sonra en küçüğü yapay sinir hücresinin net girdisi olarak kabul edilir.
Çoğunluk $Net\ girdi = \sum_i \text{sgn}(w_i x_i)$	N adet girdi içinden ağırlıklar ile çarpıldıktan sonra pozitif ve negatif olanların sayısı bulunur. Büyük olan sayı hücrenin net girdisi olarak kabul edilir.
Kümülatif toplam $Net\ girdi = Net_i(eski) + \sum w_i x_i$	Hücreye gelen bilgiler ağırlıklı olarak toplanır ve daha önce gelen bilgilere eklenerek hücrenin net girdisi bulunur.

Aktivasyon fonksiyonu, hücreye gelen net girdiyi işleyerek hücrenin bu girdiye karşılık üreteceği net çıktıyı belirler. Toplama fonksiyonunda olduğu gibi aktivasyon fonksiyonu olarak da değişik formüller kullanılmaktadır. Bazı modeller (çok katmanlı algılayıcı modeli gibi) bu fonksiyonun türevi alınabilir bir fonksiyon olmasını şart koşmaktadır. Yapay sinir ağı modllerinde ağın işlem elemanlarının hepsinin aynı aktivasyon fonksiyonu kullanması şart değildir, farklı fonksiyonlar da kullanılabilir. Bir problem için en uygun fonksiyonun belirlenmesi tasarımcının denemeleri sonucunda belirleyebileceği bir durumdur. Uygun fonksiyonu gösteren bir formül bulunmuş değildir [115]. Günümüzde en yaygın olarak sigmoid aktivasyon fonksiyonu kullanılmaktadır [73] .

Çıktı, aktivasyon fonksiyonu tarafından belirlenen çıktı değeridir. Üretilen çıktı başka bir hücreye ya da dış dünyaya verilebilir. Bazı yapılarda hücre kendi çıktısını kendisine de yollayabilir.

3.8. Yapay Sinir Ağlarının Yapısı

Nöronlar bir grup halinde işlem gördüklerinde bu gruba ağ denilir ve bir ağda binlerce nöron bulunur. Yapay nöronların birbirleriyle bağlantılar aracılığıyla bir araya gelmeleri yapay sinir ağını oluşturur. Yapay sinir ağında, nöronların aynı doğrultu üzerinde bir araya gelmeleriyle katmanlar oluşur. Katmanların farklı bağlantıları değişik ağ yapılarını ortaya çıkarır. Yapay sinir ağları üç katman şekli içermektedir. Bunlar;

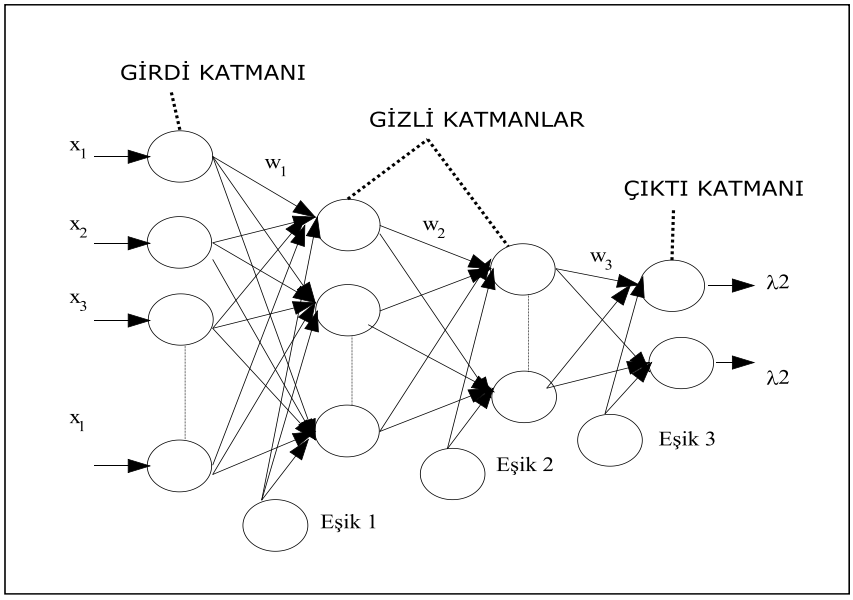
- Girdi katmanı
- Gizli katman(lar)
- Çıktı katmanıdır.

Girdi katmanı: Girdi katmanındaki işlem elemanları dış dünyadan bilgileri alarak gizli katmanlara taşırlar. Bazı ağlarda girdi katmanında herhangi bir bilgi işleme olmaz.

Gizli katman(lar): Gizli katman veya katmanlarda bulunan işlem elemanları yani nöronlar girdi katmanından gelen bilgileri işleyerek çıktı katmanına gönderirler.

Çıktı katmanı: Bu katmandaki işlem elemanları gizli katman(lar)dan gelen bilgileri işleyerek ağın girdi katmanından sunulan girdi kümesi için üretmesi gereken çıktıyı üretirler. Üretilen çıktı dış dünyaya gönderilir.

Bu üç katmanın her birinde bulunan nöronlar ve katmanlar arası ilişkiler şematik olarak Şekil 3.4.'de gösterilmektedir.



Şekil 3.4. Yapay sinir ağı modeli

3.9. Aktivasyon Fonksiyonları

Transfer fonksiyonu veya öğrenme eğrisi olarak da adlandırılan aktivasyon fonksiyonları bir yapay sinir ağında nöronun çıkış genliğini istenilen değerler arasında sınırlar. Bu değerler çoğunlukla [0, 1] veya [-1, 1] arasındadır. Yapay sinir ağlarında kullanılacak olan fonksiyonların türevi alınabilir ve süreklilik arz eden fonksiyonlar olması gerekir. Amaca göre tek veya çift yönlü aktivasyon fonksiyonları da kullanılabilir. Uygulamalarda daha çok sigmoid veya hiperbolik tanjant aktivasyon fonksiyonları kullanılmaktadır [119]. Örneğin eğer ağın bir modelin ortalama davranışını öğrenmesi isteniyorsa sigmoid fonksiyon, ortalamadan sapmanın öğrenilmesi isteniyorsa hiperbolik tanjant fonksiyon kullanılması önerilmektedir.

Doğrusal veya doğrusal olmayan aktivasyon fonksiyonlarının kullanılması yapay sinir ağlarının karmaşık ve çok farklı problemlere uygulanmasını sağlamıştır. En çok kullanılan aktivasyon fonksiyonları izleyen bölümde detaylı olarak açıklanmıştır.

Doğrusal Aktivasyon Fonksiyonu:

Doğrusal bir problemi çözmek amacıyla kullanılan doğrusal hücre ve genellikle sinir ağlarının çıktı katmanında kullanılan doğrusal fonksiyon, hücrenin net girdisini doğrudan hücre çıkışı olarak verir. Genellikle ADALINE olarak isimlendirilen doğrusal

işlemci eleman matematiksel olarak (3.4.) eşitliği ile gösterilir. Burada *A*, sabit bir katsayıdır.

$$y = Av \tag{3.4.}$$

Yapay sinir ağlarının çıkış katmanında kullanılan doğrusal fonksiyon Şekil 3.5.(a)'da gösterilmektedir.

Basamak Fonksiyonu:

Algılayıcı olarak bilinen işlemci eleman bu fonksiyon ile işlem görür. Basamak fonksiyonu tek veya çift kutuplu fonksiyon olabilir. Matematiksel modeli (3.5.) ve (3.6.)'da verilmektedir. Basamak fonksiyonun davranışı Şekil 3.5.(b)'de gösterilmektedir.

$$y = F(v) = \begin{cases} 1, & v \geq 0 \\ 0, & x < 0 \end{cases} \tag{3.5.}$$

$$y = F(v) = \begin{cases} 1, & v \geq 0 \\ -1, & v < 0 \end{cases} \tag{3.6.}$$

Sigmoid Aktivasyon Fonksiyonu:

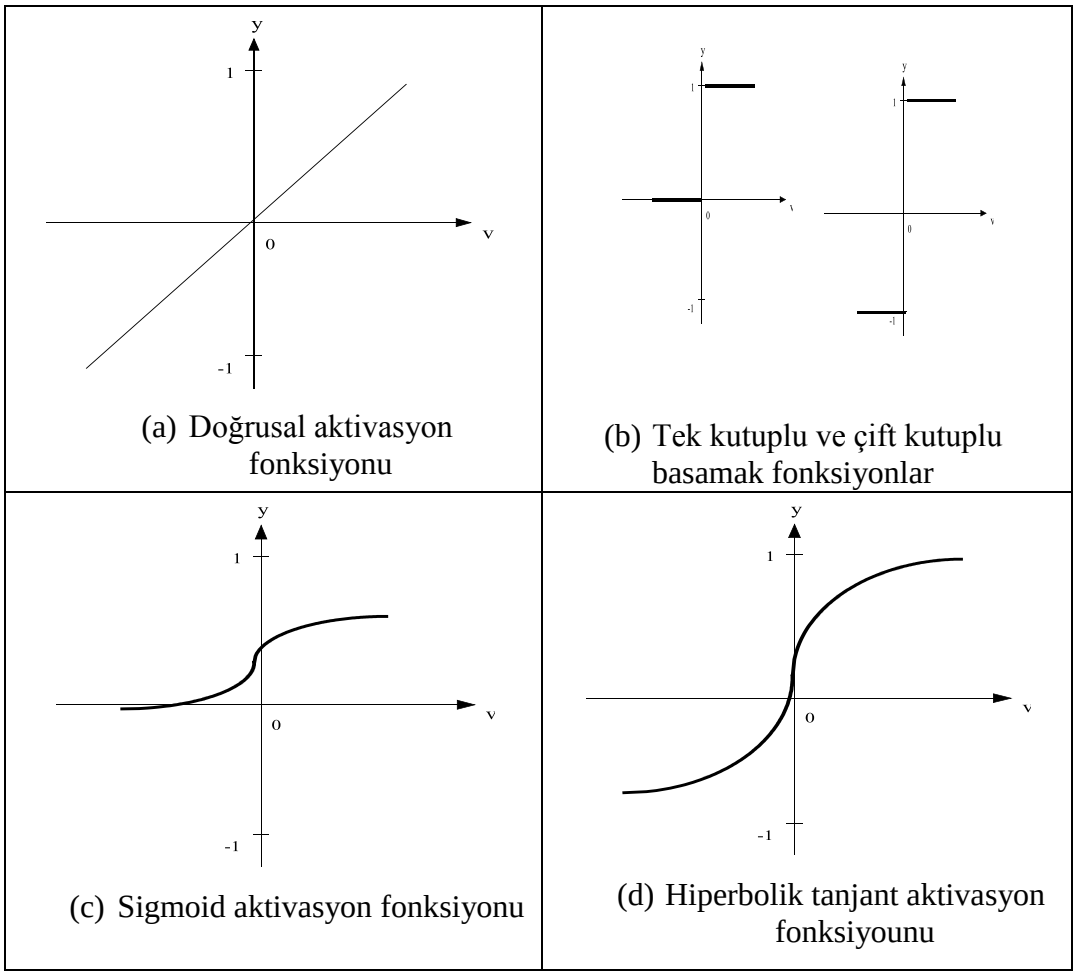
Sigmoid aktivasyon fonksiyonu, türevi alınabilir, sürekli ve doğrusal olmayan bir fonksiyon olması nedeniyle uygulamada en çok kullanılan aktivasyon fonksiyonudur. Bu fonksiyon girdinin her değeri için 0 ile 1 arasında değerler üretir. Fonksiyonun davranışı Şekil 3.5.(c)'de, formülü ise eşitlik (3.7.)'de verilmektedir [121].

$$y = \frac{1}{1+e^{-v}} \tag{3.7.}$$

Hiperbolik Tanjant Aktivasyon Fonksiyonu:

Hiperbolik tanjant fonksiyonu, sigmoid fonksiyonunun farklı bir şeklidir. Giriş uzayının genişletilmesinde etkili bir aktivasyon fonksiyonudur. Sigmoid fonksiyonu [0,1] aralığında değer alırken, hiperbolik tanjant fonksiyonu [-1, 1] aralığında değer alır. Şekil 3.5.(d) hiperbolik tanjant fonksiyonun grafiğini göstermektedir. Formülü ise (3.8.)'de verilmiştir [121];

$$y = \frac{1-e^{-2v}}{1+e^{2v}} \qquad (3.8.)$$



Şekil 3.5. Aktivasyon fonksiyonları.

Anlatılan bu aktivasyon fonksiyonlarından başka literatürde geçen diğer aktivasyon fonksiyonları ise şöyle sıralanabilir;

- Parçalı doğrusal aktivasyon fonksiyon
- Uyarlanabilir parametreli aktivasyon fonksiyon

3.10. Yapay Sinir Ağlarının Sınıflandırılması

Yapay sinir ağları, birbirleri ile bağlantılı nöronlardan oluşurlar. Her bir nöron arasındaki bağlantıların yapısı ağın yapısını belirler. İstenilene ulaşmak için bağlantıların nasıl değiştirileceği öğrenme algoritması tarafından belirlenir. Buna göre yapay sinir ağları, yapılarına ve öğrenme algoritmalarına göre sınıflandırılabilirler.

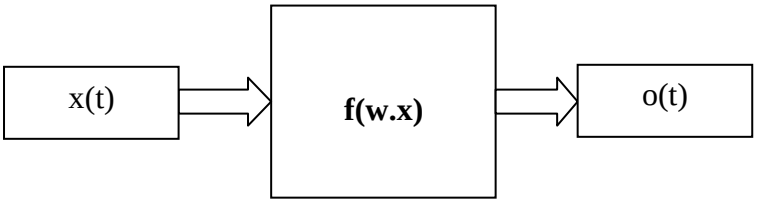
- YSA'nın yapılarına göre sınıflandırılması
    - İleri beslemeli ağlar
    - Geri beslemeli (recurrent) ağlar

- YSA'nın öğrenme algoritmalarına göre sınıflandırılması
    - Danışmanlı öğrenme
    - Danışmansız öğrenme
    - Takviyeli öğrenme
    - Karma stratejiler

3.10.1. Yapılarına Göre Sınıflandırma

Yapay sinir ağları yapılarına göre ileri beslemeli ve geri beslemeli ağlar olmak üzere ikiye ayrılırlar.

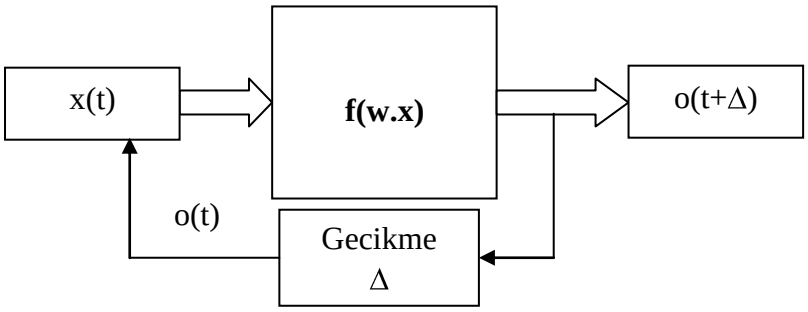
İleri beslemeli ağlar: İleri beslemeli ağlarda nöronlar, genellikle katmanlara ayrılır. Bilgi akışı giriş katmanından çıkış katmanına doğru tek yönlü bağlantılarla sağlanır. Bu ağlarda gecikme olmaz ve nöronlar bir katmandan diğer bir katmana bağlantı kurarlarken, aynı katman içerisinde bağlantıları bulunmaz. İleri beslemeli ağlarda çıkış değerleri istenen çıkış değerleri ile karşılaştırılarak bir hata sinyali ile ağ ağırlıkları güncellenir. Şekil 3.6. ileri beslemeli ağlar için blok diyagramını göstermektedir. İleri beslemeli ağlara örnek olarak çok katmanlı algılayıcı modeli verilebilir.



Şekil 3.6. İleri beslemeli YSA blok diyagramı.

Geri beslemeli ağlar: Geri beslemeli YSA'da, en az bir hücrenin çıkışı kendisine veya diğer hücrelere giriş olarak verilir ve genellikle geri besleme bir geciktirme elemanı üzerinden yapılır. Geri besleme bir katmandaki hücreler arasında olduğu gibi, katmanlar arasındaki hücreler arasında da olabilir. Bu yapısı ile geri beslemeli ağ, doğrusal olmayan dinamik bir davranış gösterir. Dolayısıyla, geri beslemenin yapılış şekline göre

farklı yapıda ve davranışta geri beslemeli YSA yapıları elde edilebilir. Ayrıca geri beslemeli ağlarda gecikmeler söz konusudur ve ağ yapısı çıkışlar girişlere bağlanarak ileri beslemeli ağlardan elde edilir. Ağın (t) anındaki çıkışı o(t) ise (t+Δ) anındaki çıkışı ise o(t+Δ)'dır. Buradaki Δ sembolik anlamda gecikme süresidir. Bu ağlara örnek olarak Hopfield, Elman ve Jordan ağları verilebilir.

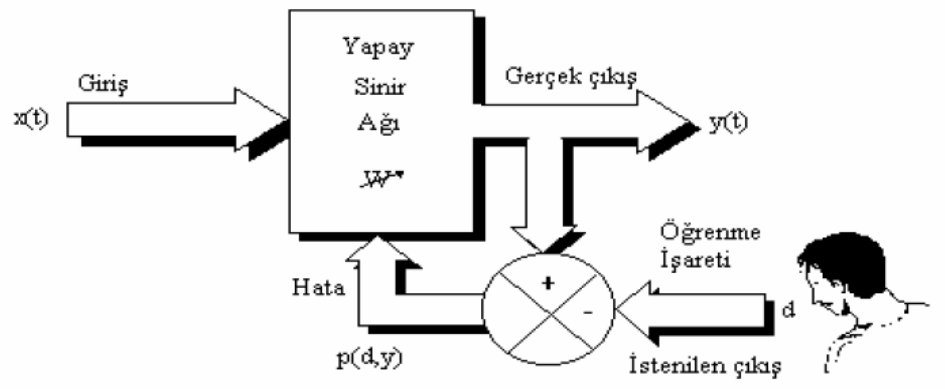


Şekil 3.7. Geri beslemeli YSA blok diyagramı.

3.10.2. Öğrenme Algoritmalarına Göre Sınıflandırma

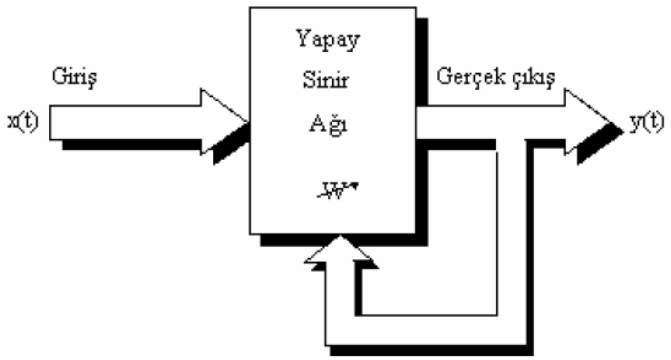
Örneklerden öğrenen yapay sinir ağlarında değişik öğrenme stratejileri kullanılmaktadır. Öğrenmeyi gerçekleştirecek olan sistem ve kullanılan öğrenme algoritması bu stratejilere bağlı olarak değişmektedir. Zurada [122], öğrenme stratejilerini danışmanlı ve danışmansız öğrenme olarak ikiye ayırsa da genel olarak aşağıdaki dört öğrenme stratejisi kullanılmaktadır.

Danışmanlı öğrenme: Danışmanlı öğrenmede sinir ağına örnek olarak bir doğru çıkış verilir. Bu öğrenmede ağın ürettiği çıktılar ile hedef çıktılar arasındaki fark hata olarak ele alınır ve bu hata en küçüklenmeye çalışılır. Bunun için de bağlantıların ağırlıkları en uygun çıkışı verecek şekilde değiştirilir. Bu sebeple danışmanlı öğrenme algoritmasının bir danışmana ihtiyacı vardır. Bu öğrenme modelinde giriş ve çıkış örnekleri kümesi "eğitim kümesi" olarak adlandırılır. Şekil 3.8.'de danışmanlı öğrenme yapısı gösterilmektedir. Çok katmanlı algılayıcı ağı danışmanlı öğrenme algoritmalarına örnek olarak verilebilir.



Şekil 3.8. Danışmanlı öğrenme yapısı.

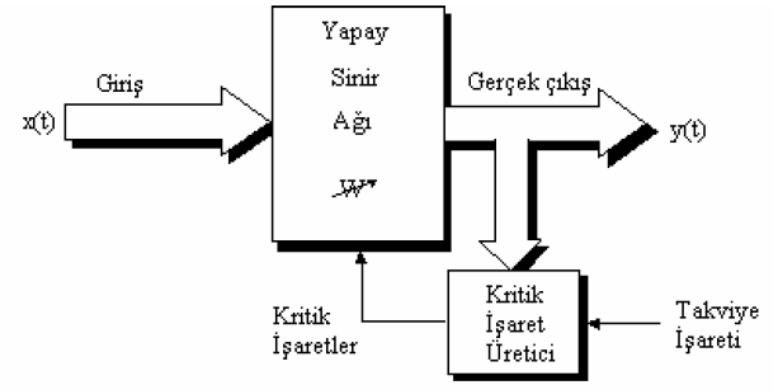
Danışmansız öğrenme: Danışmansız öğrenmede ağa sadece girdiler verilir. Ağın ulaşması gereken hedef çıktılar verilmez. Girişe verilen örnekten elde edilen çıkış bilgisine göre ağ sınıflandırma kurallarını kendi kendine geliştirir. Ağ daha sonra bağlantı ağırlıklarını aynı özellikleri gösteren desenler oluşturmak üzere ayarlar. Şekil 3.9. danışmansız öğrenme yapısını göstermektedir. Bu tip öğrenmeye adaptif rezonans teori (ART) ve özdüzenleyici haritalar örnek olarak verilebilir.



Şekil 3.9. Danışmansız öğrenme yapısı.

Takviyeli öğrenme: Takviyeli öğrenmede öğrenen sisteme bir danışman yardımcı olur. Fakat öğretmen her girdi seti için olması gereken çıktı setini sisteme göstermek yerine sistemin kendisine gösterilen girdilere karşılık çıktısını üretmesini bekler ve üretilen çıktının doğru veya yanlış olduğunu gösteren bir sinyal üretir. Sistem danışmandan gelen bu sinyali dikkate alarak öğrenme sürecini devam ettirir. Şekil 3.10. takviyeli

öğrenme yapısını göstermektedir. Boltzmann kuralı veya GA takviyeli öğrenmeye örnek olarak verilebilir.



Şekil 3.10. Takviyeli öğrenme yapısı.

Karma stratejiler: Yukarıdaki üç stratejiden birkaçını birlikte kullanarak öğrenme gerçekleştiren ağlarda vardır. Bu ağlar, kısmen danışmanlı kısmen de danışmansız olarak öğrenme gerçekleştirmektedirler. Radyal tabanlı yapay sinir ağları ve olasılık tabanlı ağlar bunlara örnek olarak verilebilir.

3.11. Temel Öğrenme Kuralları

Yapay sinir ağlarında öğrenme, danışmanlı, danışmansız, takviyeli veya karma stratejilerden hangisi uygulanırsa uygulansın bazı kurallara göre gerçekleştirilmesi gerekir. Öğrenme işlemi çok parametreli, karmaşık ve matematiksel olarak ifade edilmesi zor bir işlemdir. Bugün kullanılan öğrenme kuralları bu işlemin basitleştirilmiş veya farklı şekilde ifade edilmiş biçimleridir [119]. Literatürde mevcut öğrenme algoritmalarının çoğu Hebb, Delta, Kohonen ve Hopfield olmak üzere dört farklı öğrenme kuralından esinlenilerek geliştirilmiştir. Bu kurallar izleyen bölümde açıklanmaktadır;

Hebb kuralı: Hebb [123] tarafından geliştirilen ve diğer öğrenme kurallarının temelini oluşturan Hebb kuralı, bilinen en eski öğrenme kuralıdır. Bu kuralın temelinde, eğer bir yapay sinir ağı elemanı diğer bir elemandan bilgi alırsa ve her iki yapay eleman da aktif ise (matematiksel olarak aynı işareti taşıyorsa) iki yapay sinir ağı elemanı arasındaki bağlantı kuvvetlendirilmelidir düşüncesi vardır. Bu kural şu şekilde özetlenebilir; bir

yapay eleman kendisi aktif ise bağlı olduğu yapay sinir ağı elemanını aktif yapmaya, pasif ise pasif yapmaya çalışmaktadır.

Hopfield kuralı: Bu kural, zayıflatma ve kuvvetlendirme büyüklüğü dışında Hebb kuralına benzemektedir. Kuralda yapay sinir ağı bağlantılarının ne kadar kuvvetlendirilmesi veya zayıflatılması gerektiği belirlenir. Eğer beklenen çıktı ve girdiler ikisi de aktif/pasif ise öğrenme katsayısı kadar ağırlık değerleri kuvvetlendirilir/zayıflatılır. Yani, ağırlıkların kuvvetlendirilmesi veya zayıflatılması öğrenme katsayısı yardımı ile gerçekleştirilmektedir. Öğrenme katsayısı genel olarak 0-1 arasında kullanıcı tarafından belirlenen sabit ve pozitif bir değerdir.

Delta kuralı: Hebb kuralının değişik bir formudur ve en çok kullanılan öğrenme algoritmalarından birisidir. Beklenen çıktı ile gerçekleşen çıktı arasındaki farklılığı azaltmak için yapay sinir ağının elemanlarının bağlantılarına yönelik ağırlık değerlerinin sürekli değiştirilmesi ilkesine dayanarak geliştirilmiştir. Ağın ürettiği çıktı ile üretilmesi gereken yani beklenen çıktı arasındaki hatanın karelerinin ortalamasını en küçüklemek amaçlanmaktadır. Hata, aynı anda bir katmandan önceki katmanlara geri yayılarak azaltılır. Ağın hatalarının düşürülmesi işlemi, çıkış katmanından giriş katmanına ulaşıncaya kadar devam eder. Bu kural, geri yayılım, Widrow-Hoff veya en küçük ortalama karesel öğrenme kuralı olarak da adlandırılır.

Kohonen kuralı: Biyolojik sistemlerdeki öğrenmeden esinlenen Kohonen [124] tarafından geliştirilen bu kuralda, nöronlar öğrenmek için yarışırlar ve kazanan nöronun ağırlıkları güncellenir. Bu kural "kazanan hepsini alır" olarak da adlandırılır. En büyük çıkışa sahip işlemci nöron kazanır. Bu, o nöronun çevresindeki nöronlara karşı daha kuvvetli hale gelmesi demektir. Hem kazanan nöronun, hem de komşuları sayılan nöronların ağırlıklarını değiştirmesine izin verilir. Kohonen kuralı hedef çıkışa ihtiyaç duymadığından danışmansız bir öğrenme metodudur.

3.12. Temel Öğrenme Algoritmaları

Geriyayılım Algoritması: Genelleştirilmiş delta algoritması olarak da isimlendirilen geri yayılım algoritması ileri beslemeli ve çok katmanlı bir ağ mimarisi gerektirmektedir. Anlaşılması kolay ve matematiksel olarak kolayca ispatlanabilir olmasından dolayı en çok tercih edilen öğrenme algoritmasıdır. Geriyayılım algoritması olarak

isimlendirilmesinin nedeni hataları çıkıştan girişe yani geriye doğru azaltmaya çalışmasıdır. Çok katmanlı algılayıcıları eğitmekte daha çok bu algoritma kullanılmaktadır [53]. Temelde istenilen çıkış ile ağ çıkışı arasındaki hatanın ağırlıklara bağlı olarak düşürülmesi prensibine dayanmaktadır. Geriyayılım algoritmasında örnekler ağa öğretilir ve ağa hedef değeri verilir. Öğrenme esnasında, her örnek için ağın çıktı değeri ile hedef değeri karşılaştırılır. Hata değeri, ağa tekrar geri besleme şeklinde iletilir. Eğitim kümesindeki hata kareleri toplamını en küçüklemek için nöronlar arasındaki bağlantı ağırlıkları değiştirilir [125]. Tipik çok katmanlı geri yayılım ağı, daima bir giriş, bir çıkış ve en az bir gizli katman içerir. Öğrenme algoritması olarak geriyayılım algoritması kullanıldığında iki parametre önem kazanır. Bunlar öğrenme katsayısı (η) ve momentum katsayısıdır (α). Öğrenme katsayısı, ağırlıkların bir sonraki iterasyonda hangi oranda değiştirileceğini göstermektedir. Küçük öğrenme katsayıları, ağın sonuca ulaşmasını yavaşlatırken, büyük öğrenme katsayıları ağın sonuca daha kısa sürede ulaşmasını sağlar. Bununla birlikte çok yüksek oranlar ağın hesaplamalarında büyük salınımlara neden olur ve ağın optimum noktayı bulmasını engelleyebilir. Momentum katsayısı ise, ağdaki salınımları engellemeye ve ağın hata yüzeyindeki bölgesel minimum noktalardan kaçarak, daha optimum noktalara ulaşmasına yardımcı olur. Optimal öğrenme oranı ve momentum katsayısının belirlenmesi büyük ölçüde deneysel ve sezgisel bir özellik taşır.

Delta-Bar-Delta: Çok katmanlı algılayıcılarda bağlantı ağırlıklarının yakınsama hızını artırmak için kullanılan sezgisel bir yaklaşımdır. Hata yüzeyindeki değişiklikleri açıklamak için, özellikle ağın her bağlantısı kendi öğrenme katsayısına sahip olmalıdır [126]. Öğrenme katsayısı artırma ve azaltma faktörü, öğrenme katsayısı temel parametreleridir. Delta-bar-delta algoritması, öğrenme katsayılarını doğrusal olarak artırmakta fakat geometrik olarak azaltmaktadır.

Hızlı yayılım algoritması: Hızlı yayılım algoritması Fahlman [127] tarafından geliştirilen ve Newton metoduna dayanan, çok katmanlı algılayıcıların eğitilmesinde kullanılan sezgisel bir algoritmadır. Hızlı yayılım algoritması, her bir ağırlık için, ağırlık hata eğrisi kolları yukarı doğru açık olan bir parabol ile yaklaştırılabilir ve hata eğrisinin eğilimindeki değişim, diğer tüm ağırlıkların aynı andaki değişimlerinden etkilenmez varsayımlarına dayanır. Genellikle performans olarak oldukça iyi sonuçlar veren bir algoritmadır.

Eşleştirmeli eğim: Eşleştirmeli eğim algoritmasında, eğim azaltım yöntemindeki doğrultulardan genellikle daha hızlı yakınsayan eşleştirme doğrultularında bir arama işlemi yapılır [128]. Eğim azaltımlı öğrenme algoritmalarında küresel minimuma doğru olan ağırlık güncellemesindeki adım büyüklüğünü belirleyen bir öğrenme oranı parametresi kullanılmaktadır. Eşleştirmeli eğim algoritmasında adım boyutu, her bir iterasyonda yeniden güncellenir. Performans fonksiyonun o doğrultu boyunca en küçükleneceği adım boyutunu belirlemek için eşleştirmeli eğim doğrultusu boyunca bir arama gerçekleştirilir [129]. Literatürde farklı eşleştirmeli eğim algoritmaları mevcuttur. Bunlara örnek olarak Fletcher-Reeves, Powell-Beale ve ölçeklendirilmiş eşleştirmeli eğim verilebilir.

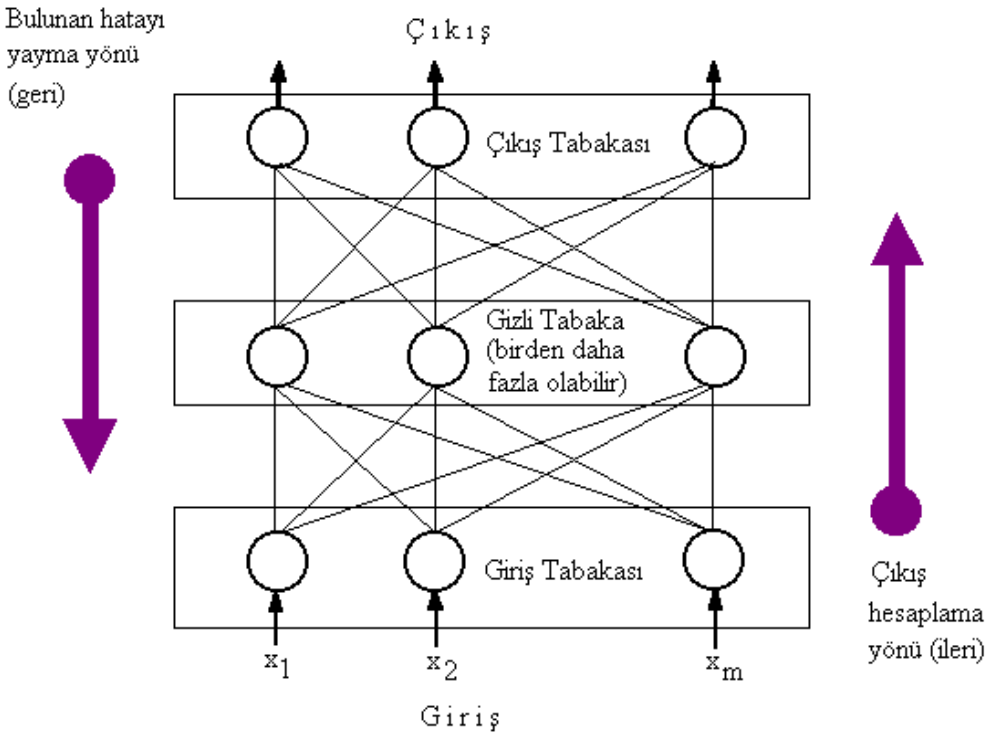
Yukarıda bahsedilen öğrenme algoritmalarının haricinde literatürde çeşitli öğrenme algoritmaları mevcuttur. Bunlardan bazıları aşağıda sıralanmıştır.

- Esnek yayılım algoritması
- Geliştirilmiş delta-bar-delta
- Genetik algoritma
- Yönlendirilmiş rastgele arama
- Kuasi-Newton
- BFGS (Broyden-Fletcher-Goldfarb-Shanno)
- Tek adım sekant öğrenme algoritması.

3.13. Çok Katmanlı Algılayıcı

XOR problemini çözmek amacı ile yapılan çalışmalar sonucu çok katmanlı algılayıcı modeli geliştirilmiştir. Rumelhart ve arkadaşları [130] tarafından geliştirilen bu modele hata yayma modeli veya geriyayılım modeli de denilmektedir. Bu model günümüzde mühendislik problemlerinin hemen hemen hepsine çözümler üretebilecek bir güce sahiptir ve en popüler yapay sinir ağı modellerinden biridir [131]. Özellikle sınıflandırma, fonksiyon tahminleme ve uzman sistemler için çok önemli bir çözüm aracıdır [132]. ÇKA modeli, delta öğrenme kuralını kullanmaktadır. Temel amacı ağın beklenen çıktısı ile ürettiği çıktı arasındaki hatayı en aza indirmektir. Bunu hatayı ağa yayarak gerçekleştirdiği için bu ağa hata yayma ağı da denilmektedir.

Bir ÇKA modeli, bir giriş, bir veya daha fazla gizli ve bir de çıkış katmanından meydana gelir. Örnek bir ÇKA yapısı Şekil 3.11.'de gösterilmektedir. Bir katmandaki bütün işlem elemanları bir üst katmandaki bütün işlem elemanlarına bağlanmıştır. Bilgi akışı ileri doğru olup geri besleme yoktur. Bunun için ileri beslemeli sinir ağı modeli olarak da adlandırılır. Giriş katmanında herhangi bir bilgi işleme yapılmaz. Buradaki işlem elemanı yani nöron sayısı tamamen uygulanan problemin giriş sayısına bağlıdır. Gizli katman sayısı ve gizli katmanlardaki işlem elemanı sayısı ise deneme-yanılma yolu ile bulunur. Çıkış katmanındaki eleman sayısı ise yine uygulanan problemdeki çıkış sayısına göre belirlenir.



Şekil 3.11. Çok katmanlı algılayıcı yapısı.

Çok katmanlı algılayıcı ağı danışmanlı öğrenme stratejisi kullanmaktadır. Dolayısıyla ağa hem örnekler hem de örneklerden elde edilmesi gereken çıktılar (beklenen çıktı) verilmektedir. Ağ, kendisine gösterilen örneklerden genelleme yaparak problem uzayını temsil eden bir çözüm uzayı üretmektedir. Daha sonra gösterilen benzer örnekler için bu çözüm uzayı, sonuçlar ve çözümler üretebilmektedir.

3.13.1. Çok Katmanlı Algılayıcı Ağının Öğrenme Kuralı

ÇKA ağlarına eğitim sırasında hem girdiler hem de girdilere karşılık üretilmesi gereken çıktılar gösterilir (danışmanlı öğrenme). Ağın görevi her girdi için o girdiye karşılık gelen çıktıyı üretmektir. Öğrenme kuralı olarak en küçük kareler yöntemine dayanan delta öğrenme kuralının genelleştirilmiş halini kullanmaktadır. Ağın öğrenebilmesi için eğitim kümesi olarak isimlendirilen ve örneklerden oluşan bir kümeye ihtiyaç vardır. Bu küme içinde her örnek için ağın hem girdiler hem de o girdiler için ağın üretmesi gereken çıktılar belirlenmiştir. Genelleştirilmiş delta kuralı iki aşamadan oluşur;

- İleri doğru hesaplama: ağın çıktısının hesaplanması aşamasıdır.
- Geriye doğru hesaplama: ağırlıkların değiştirilme aşamasıdır.

İleri doğru hesaplama: ÇKA'da bilgi işleme eğitim kümesindeki bir örneğin girdi katmanından ağa gösterilmesi ile başlar ancak girdi katmanında herhangi bir bilgi işleme olmaz, gelen girdiler hiçbir değişiklik olmadan ara katmana gönderilir. Girdi katmanındaki i girdi katmanını ifade etmek üzere, *l.* işlem elemanının çıktısı O_l^i, eşitlik (3.9.) ile hesaplanır. Gizli katmanlardaki işlem elemanları girdi katmanındaki tüm işlem elemanlarından gelen bilgileri bağlantı ağırlıkları (w_{lj} l=1….girdi proses elemanı sayısı, *j*=1…..gizli katman proses eleman sayısı) etkisi ile alırlar. *a*, gizli katmanı ifade etmek üzere, gizli katmandaki işlem elemanlarına gelen net girdi toplama fonksiyonu ile (3.10.) eşitliği ile hesaplanır. *j.* gizli katmanın çıktısı ise (3.10.) eşitliği ile hesaplanan net girdinin aktivasyon fonksiyonundan geçirilmesi ile hesaplanır. Sigmoid aktivasyon fonksiyonu kullanılması halinde çıktı (3.11.) eşitliği ile hesaplanır. Burada β_j^a, gizli katmanın *j.* elemanına bağlanan eşik değer elemanını göstermektedir. Gizli katmanın bütün işlem elemanları ve çıktı katmanının işlem elemanlarının çıktıları aynı şekilde kendilerine gelen NET girdinin hesaplanması ve sigmoid aktivasyon fonksiyonundan geçirilmesi vasıtasıyla belirlenir. Çıktı katmanından çıkan değerler (çıktılar) bulununca, ağın ileri doğru hesaplama işlemi tamamlanmış olur.

$$O_l^i = I_l \tag{3.9.}$$

$$NET_j^a = \sum_{l=1}^{n} w_{lj}\, O_j^i \tag{3.10.}$$

$$O_j^a = \frac{1}{1+e^{-(NET_j^a+\beta_j^a)}} \quad (3.11.)$$

Geriye doğru hesaplama: Ağın ürettiği hata değeri gerçekleşen ile beklenen çıktı kullanılarak hesaplanır. Amaç bu hatanın düşürülmesidir. Bu nedenle, geriye doğru hesaplama yapılırken bu hata ağın ağırlık değerlerine dağıtılarak bir sonraki iterasyonda hatanın azaltılması sağlanır. Çıktı katmanındaki *k.* işlem elemanı için oluşan hata (3.12.) eşitliği ile hesaplanır [122, 133]. Burada d_k beklenen çıktıyı, o_k ise gerçekleşen çıktıyı ifade etmektedir. Bu bir işlem elemanı için oluşan hatadır. Çıktı katmanında oluşan toplam hatayı bulmak için (3.13.) eşitliği kullanılır [53].

$$E_k = (d_k - o_k) \quad (3.12.)$$

$$E(t) = \frac{1}{2}\sum_{k=1}^{K} E_k^2 \quad (3.13.)$$

Üretilen hatanın en küçüklenmesi için ağırlıklar geriye doğru hesaplanarak değiştirilir. Bunun için hata, kendisine neden olan işlem elemanlarına dağıtılır. Ağın ağırlıklarını değiştirmek için iki durum söz konusudur: gizli katman-çıktı katmanı arasındaki ağırlıkların değiştirilmesi, gizli katmanlar arası veya gizli katman-girdi katmanı arasındaki ağırlıkların değiştirilmesi. λ öğrenme katsayısını, α momentum katsayısını, δ_k k. Çıktı ünitesinin hatasını göstermek üzere, gizli katmandaki *j.* işlem elemanını çıktı katmanındaki *k.* işlem elemanına bağlayan bağlantının ağırlığındaki değişim miktarı ΔD^a (3.14.) eşitliği ile hesaplanır. Ağırlıkların *t.* iterasyondaki yeni değerleri (3.15.) eşitliği ile hesaplanır. Eşik değer ünitesi için de benzer hesaplamalar (3.16.) ve (3.17.) formülleri ile yapılır. Gizli katmanlar arası veya gizli katman-girdi katmanı arasındaki ağırlıkların değiştirilmesi ise benzer şekilde (3.18.) ve (3.19.) eşitlikleri ile sağlanır. Burada bulunan eşik değer ünitesinin yeni ağırlıkları da (3.20.) ve (3.21.) formülleri ile hesaplanır.

$$\Delta D_{jk}^a(t) = \lambda \delta_k O_j^a + \alpha \Delta D_{jk}^a(t-1) \quad (3.14.)$$

$$D_{jk}^a(t) = D_{jk}^a(t-1) + \Delta D_{jk}^a(t) \quad (3.15.)$$

$$\Delta \beta_k^c(t) = \lambda \delta_k + \alpha \Delta \beta_k^c(t-1) \quad (3.16.)$$

$$\beta_k^c(t) = \beta_k^c(t-1) + \Delta\beta_k^c(t) \quad (3.17.)$$

$$\Delta D_{lj}^i(t) = \lambda \delta_j^a O_l^i + \alpha \Delta D_{yj}^i(t-1) \quad (3.18.)$$

$$D_{lj}^i(t) = D_{lj}^i(t-1) + \Delta D_{lj}^i(t) \quad (3.19.)$$

$$\Delta\beta_j^a(t) = \lambda \delta_j^a + \alpha \Delta\beta_j^a(t-1) \quad (3.20.)$$

$$\beta_j^a(t) = \beta_j^a(t-1) + \Delta\beta_j^a(t) \quad (3.21.)$$

3.13.2. Çok Katmanlı Algılayıcı Ağının Çalışması

Çok katmanlı algılayıcılar aşağıdaki adımları gerçekleştirerek çalışmaktadırlar. Bu adımlar ÇKA ağının öğrenmesi tamamlanıncaya kadar, yani gerçekleşen çıktılar ile beklenen çıktılar arasındaki hatalar kabul edilebilir düzeye ininceye kadar devam ettirilir. Ağın öğrenmesi için bir durdurma kriterinin olması gerekmektedir. Bu ise genellikle üretilen hatanın belirli bir düzeyin altına düşmesi veya belirli bir iterasyon sayısının tamamlanması olarak ele alınmaktadır.

- Örneklerin toplanması
- Ağın topolojik yapısının belirlenmesi
- Öğrenme parametrelerinin belirlenmesi
- Ağırlıkların başlangıç değerlerinin atanması
- Eğitim kümesinden örneklerin seçilmesi ve ağa gösterilmesi
- Öğrenme sırasında ileri hesaplama yapılması
- Gerçekleşen çıktı ile beklenen çıktının karşılaştırılması
- Ağırlıkların değiştirilmesi

3.14. Diğer Yapay Sinir Ağları

LVQ Ağı: LVQ ağı, giriş ve gizli katman arasında tamamen, gizli ve çıkış katmanı arasında da kısmen bağlıdır. Her çıkış işlem elemanı farklı bir gizli işlem eleman kümesine bağlıdır. Gizli ve çıkış işlem elemanları arasındaki ağırlıklar 1'e sabitlenmiştir. Giriş-gizli işlem eleman bağlantılarının ağırlıkları "referans" vektörlerinin elemanlarını oluşturur (her gizli işlem elemana bir referans vektör

atanmıştır). Ağın öğretilmesi esnasında bunlar yeniden değer alırlar. Hem gizli işlem elemanları (bunlar Kohonen işlem elemanları olarak da bilinir) hem de çıkış işlem elemanları ikili çıkışa sahiptir. Ağa bir giriş deseni verildiğinde referans vektörü giriş desenine en yakın olan gizli işlem eleman kümesi "1", diğerleri "0" üretir. "1" üreten çıkış işlem elemanı giriş işaretini sınıflar ve her işlemci eleman ayrı bir sınıfa atanır.

Hopfield Ağı: Bu ağ genellikle ikili (0 veya 1) ve bipolar (+1 veya -1) girişler kabul eder. Tek tabaka işlemci elemanları vardır ve her işlemci eleman bir diğerine bağlanmıştır. Bu da daha önce belirtilen geri beslemeli yapıdır.

Elman ve Jordan Ağları: Bu ağlar, çok katmanlı algılayıcılara benzer bir yapıdadır ve çok katlıdırlar. Her iki ağda da gizli katmana ek olarak diğer bir "durum" katmanı denilen özel bir gizli katman daha vardır. Bu katman gizli katmandan veya çıkış katmanından geri besleme işaretleri alır. Jordan ağının aynı zamanda durum katmanındaki her işlem elemanından kendisine bağlantıları vardır. Her iki ağda da durum katmanındaki işlem elemanlarının çıkışları ileriye doğru gizli katmana verilmektedir. Eğer sadece ileri doğru bağlantılar göz önüne alınır ve geri besleme bağlantılarına sabit değerler verilirse, bu ağlar sıradan ileri beslemeli ağlar haline gelirler.

Kohonen Ağı: Kohonen ağı, bir giriş katmanı ve bir de çıkış katmanı olmak üzere iki katmandan meydana gelir. Çıkış katmanındaki işlem elemanları genellikle düzenli iki boyutlu aralıklar olarak düzenlenir. Çıkıştaki her işlem elemanı, bütün giriş işlem elemanlarına bağlıdır. Bağlantıların ağırlıkları verilen çıkış işlem elemanı ile ilgili olan referans vektörünün elemanlarını oluşturur.

ART Ağı: ART ağının, ART-1, ART-2, Fuzzy-ART, MART gibi değişik çeşitleri vardır. ART-2 gerçek değerler kabul etmektedir. ART-1 ağının giriş ve çıkış olmak üzere iki katmanı vardır. İki katman birbirleriyle tamamen bağlantılıdır ve bağlantılar ileri ve geri yöndedir.

3.15. Yapay Sinir Ağlarının Uygulanmasında Karşılaşılan Problemler

Yapay sinir ağlarının başarısı, uygun parametrelerin seçimine bağlıdır. Ancak uygun parametrelerin seçimi ile ilgili kesin ortaya konmuş kriterler yoktur. Dolayısıyla bu

parametrelerin seçimi YSA uygulamalarında bazı problemler ortaya çıkarmaktadır. Bunlar [119];

- Yapay sinir ağının yapısı veya mimarisinin seçimi,
- YSA giriş ve çıkış sayılarının en uygun veya en az sayıda seçimi,
- Gizli katman sayısının seçimi,
- Gizli katman veya katmanlarda bulunacak işlem elemanı sayılarının seçimi,
- Kullanılacak öğrenme algoritmasının YSA yapısına uygun olması,
- Öğrenme algoritması parametrelerinin seçimi,
- Veri kodlama yapısı,
- Veri normalizasyon yaklaşımı,
- Seçilen aktivasyon fonksiyonunun yapısı,
- Toplama fonksiyonu tipi,
- Uygun performans fonksiyonu seçimi,
- Uygun iterasyon sayısı seçimi,
- Uygun veri tipinin ve sayısın belirlenmesi vb..

3.16. Sonuç

Bu bölümde, yapay sinir ağlarından kural çıkarımına temel oluşturması için yapay sinir ağları hakkında bilgiler verilmiştir. YSA'ların tanımı ve tarihçesi verildikten sonra, YSA'ların avantaj ve dezavantajları, uygulama alanları, biyolojik sinir sistemi, yapay sinir hücresi açıklanmıştır. YSA'ların yapısı, aktivasyon fonksiyonları ve öğrenme algoritmaları gibi YSA'ların yapısal özellikleri de yine bu bölümde yer almaktadır. Son olarak ise YSA'ların uygulanmasında karşılaşılan problemlere kısaca yer verilmiştir.

Böylece yapay sinir ağlarının işleyişi ve temel yapısı üzerinden, geliştirilen algoritmanın çalışma prensibi Bölüm 6.'da daha kolay açıklanabilmiştir.

4. BÖLÜM
YAPAY SİNİR AĞLARINDAN KURAL ÇIKARIMI

4.1. Giriş

Bu bölümde yapay sinir ağlarından kural çıkarımının amacı kısaca açıklandıktan sonra kural çıkarımı hakkında detaylı bilgiler verilerek, çıkarılan kuralların taşıması gereken özelliklerden bahsedilmiştir. Daha sonra YSA'dan kural çıkarım metotları sınıflandırılmıştır. Bölümün sonunda ise YSA'lardan metasezgisel teknikler ile kural çıkaran yaklaşımlara ilişkin detaylı bir literatür verilmiştir.

4.2. Neden Yapay Sinir Ağlarından Kural Çıkarımı?

Sinir ağları en çok kullanılan yapay zeka tekniklerinden biridir. Yapay sinir ağı modeli, sınıflandırmada ve verinin tahmin edilmesinde oldukça doğru sonuçlar vermektedir. YSA metotlarının ve sistemlerinin birçok endüstri ve bilimsel alandaki başarılı uygulamaları YSA'nın kabiliyetlerini göstermektedir.

Sinir ağlarının sınıflandırma ve fonksiyon tahminleme kavramları çoğu zaman kullanıcılar için çok zor anlaşılmaktadır. Bunun sebebi, tipik sinir ağ çözümlerinin yorumlaması güç gerçek değerli parametrelerin büyük kümeleriyle karakterize edilen, birbiriyle etkileşimli doğrusal olmayan çok sayıda elemandan oluşmasıdır. Dağıtılmış iç gösterimler, bir ağın gerçekten ne öğrendiğini ve gerçek cevabı üretmek için nerede kalacağını anlamayı daha da zor hale getirmektedir [5].

Sinir ağı sistemlerinin geniş alanda işletilmesindeki temel engel doğal tanımlama yeteneklerinin olmamasıdır. Bir ağ sistemine bu tip bir yetenek vermenin bir yolu, yeteneği sunacak sistemin klasik parçasıyla birlikte bir melez sistem sağlamaktır. Ancak, eğer klasik kısım sinir ağları ile aynı kuralları kullanıyorsa, bu durumda kural çıkarım mekanizmasına ihtiyaç vardır. Bu nedenle çoğu araştırmacı, eğitilmiş ileri

beslemeli YSA'ların sembolik kurallar şeklinde gösterimini bulmaya ve bunlardan kural çıkarma teknikleri geliştirme araştırmalarına odaklanmaktadır.

Craven ve Shavlik [134], sinir ağlarından kural çıkarma problemini "eğitilmiş bir sinir ağı ve bunu eğitmekte kullanılan örnekler verildiğinde, ağın özlü ve doğru sembolik tanımının üretilmesi" olarak tanımlamıştır. Kuralların çıkarımı ile birlikte, son kullanıcı sinir ağının ne öğrendiğini ve nasıl çalıştığını anlayabilir. Sinir ağlarının iç parametrelerinden sembolik kurallar oluşturarak bilgi çıkarımı, YSA'nın bazı sınırlamalarının üstesinden gelmekte artık kabul gören bir teknik haline gelmiştir.

4.3. Kural Çıkarımı

Sinir ağları farklı uygulama alanlarına sahip olup, konuşma ve karakter tanıma gibi birçok sınıflandırma problemine sağlam çözümler üretebilmektedir. Bununla birlikte kara kutu özelliğinden dolayı bu çözümleri nasıl ürettiği anlaşılamamaktadır. Bir yapay sinir ağındaki bilgi ağ etrafına dağılmıştır ve sayısal ağırlık parametreleri formunda saklanır. Yapay sinir ağlarının çalışma prensibini anlamaya yönelik bir yaklaşım, ağırlıklandırılmış parametreli sembolik gösterimlerle ifadedir. Bilginin sembolik yorumlanması kara kutu problemini çözebilir. Bu şekilde yeniden formülasyon kural çıkarımı olarak bilinir ve ağın davranışını, karakterini ifade eder.

Kural çıkarımı ayrıca;

- Doğrulama amacıyla ağın öğrendiği kuralları denetlemek,
- Veride mevcut olabilecek yeni ilişkileri keşfetmek ve
- Çözümlerin genellenmesini iyileştirmekte kullanılabilir.

Andrews ve arkadaşları [8], eğitilmiş yapay sinir ağlarından kural çıkarımının önemini şöyle sıralamışlardır;

- Kullanıcı tarafından anlaşılırlık garantisi,
- Kritik güvenlik problem alanlarına YSA sistemlerinin uygulanması,
- Yazılım sistemlerinde doğrulama ve YSA bileşenlerinde hata ayıklama,
- YSA çözümlerinin genellemesinin iyileştirilmesi,
- Bilimsel teorilerin çıkarımı ve veri araştırımı

- Uzman sistemler için bilgi kazanımı.

Yapay sinir ağlarından kural çıkarımının başta kullanıcının yapay sinir ağlarında gizli olan bilgiyi anlaması olmak üzere birçok avantajı vardır. Bu avantajlardan başlıcaları şu şekilde sıralanabilir;

- Ağın girdi/çıktı gösterimini kurallar formunda açıklayan mekanizmanın garantisi,
- Orijinal eğitimdeki eksiklikleri belirleme yeteneği,
- Ağ performansını artıracak çıkarım için, gereksiz ağ parametrelerinin belirlenmesi,
- Veride daha önce bilinmeyen ilişkilerin analizi,
- Nedenleme ve tanımlama kabiliyetleri,
- Çapraz-gönderme ve kanıtlama kabiliyetlerini destekleme,
- Bilgi kazanımı probleminin ele alınması ve başlangıç alan bilgisinin sağlanması.

Bunlar gibi daha pek çok avantaja sahip olan yapay sinir ağlarından kural çıkarımı algoritmalarında bulunması gereken bazı özellikler vardır. Thrun [135] 1994 yılında yaptığı çalışmasında bu özelliklerin ideal olarak neler olması gerektiğini ele almıştır. Bu özellikler;

- Mimari ihtiyaç yoktur,
- Eğitim ihtiyacı yoktur,
- Doğruluk ve
- Yüksek ifade gücüdür.

Yapay sinir ağlarından çıkarılan kurallar belli bir formda olmalıdır. Kural çıkarımında genellikle "EĞER(durum veya durumlar), O HALDE (sonuç)" şeklinde önerme matematiği kullanılır. Örnek olarak aşağıdaki kural verilebilir.

Eğer A doğruysa
ve B doğruysa
ve C yanlışsa ….
o halde X de doğrudur.

Kural gösteriminde önerme matematiğine alternatif başka bir form "N içinde M" metodudur. N içinde M yöntemi gruplama yaparak kuralı ifade etmeye çalışır. Örnek olarak aşağıdaki kural verilebilir.

Eğer şu N önde gelenlerden M'i …. doğru ise
o halde ……..doğrudur.

Diğerleri kadar sık kullanılmayan ve anlaşılması zor diğer üçüncü bir kural gösterim şekli "dolaylı kural" gösterimidir. Dolaylı kurallar parçalı diskriminant fonksiyonları ifade ederler ve genellikle aşağıdaki şekilde gösterilirler.

$c_1, c_2, \ldots..,c_6 \in R$ olmak üzere
Eğer $(c_1X+c_2Y>c_3)$ ve $((c_4X+c_5Y>c_6)$ ve …….. o halde Sınıf 1

4.3.1. Çıkarılan Kuralların Taşıması Gereken Özellikler

Yapay sinir ağlarından çıkarılan kurallarda ideal olarak beklenen özellikler geçerlik, maksimum genellik, tamlık ve doğruluk olarak sıralanabilir.

Geçerlik: Kurallar, üzerinde durulmayan değişkenlerin aldığı değerler ne olursa olsun bunları içermelidir.

Maksimum genellik: Eğer kuraldaki durumlardan herhangi biri çıkarılırsa kural doğruluğunu korumaz. Bu haliyle kural maksimum geneldir denilemez.

Tamlık: Çıkarılan kurallar eksiksiz olmalıdır.

Doğruluk: Çıkarılan kurallar ele alınan veri veya örnek kümesini tam olarak yansıtmalıdır.

4.3.2. Çıkarımlar

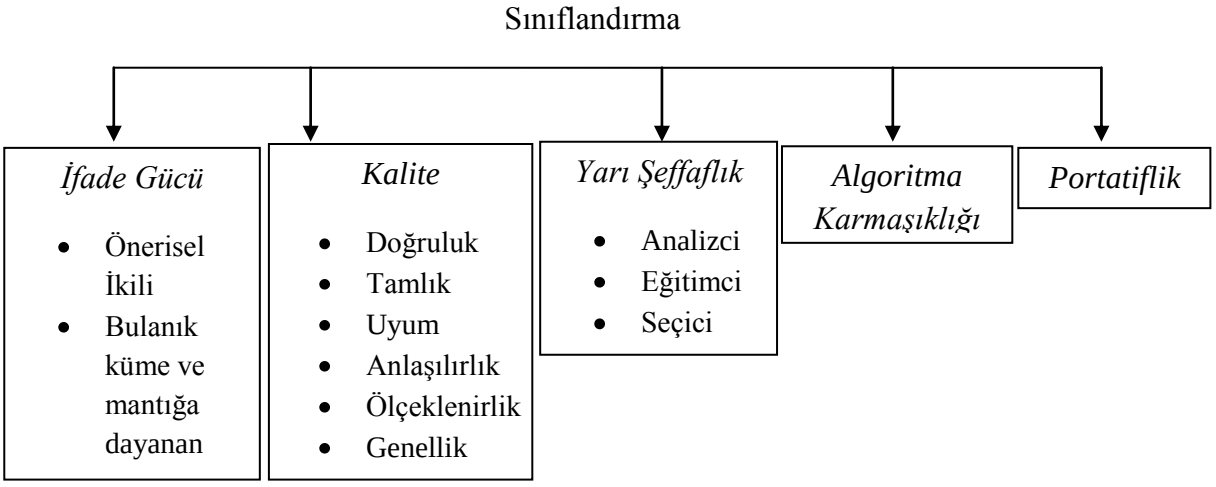
Bir kural çıkarım tekniği ile ilgili çıkarımlar şu şekilde sıralanabilir [5, 136];

- *Granürlük:* Sistemin her bir çıktı kararında sunabileceği detaylı hipotez ve ifade düzeyidir.
- *Saydamlık:* Kararların veya sonuçların ne derece iyi açıklandığıdır.

- *Genelleme yeteneği:* Kararların ve sonuçların geniş bir veri grubunu örnekleyebilmesidir.
- *Güncellenebilirlik:* İlgili eğitilmiş sinir ağ mimarisi güncellendiğinde veya farklı veri kümeleriyle yeniden eğitildiğinde çıkarılan kuralların güncellenme kabiliyetidir.
- *Sağlamlık:* Metodun, eğitim verisinde veya başlangıç alan bilgisindeki bozukluklara duyarlılığının ölçüsüdür.
- *Ölçeklenirlik:* Büyük veri kümeleri veya kural-tabanlarındaki ölçeklenirliği ifade eder.

4.3.3. YSA'dan Kural Çıkarım Tekniklerinin Sınıflandırılması

Andrews ve ark. [8], eğitilmiş sinir ağlarından kural çıkaran teknikleri sınıflandırmakta bir yaklaşım sunmuşlardır. Sınıflandırmaları ifade gücü, kalite, yarı şeffaflık, algoritma karmaşıklığı ve portatiflik olmak üzere beş kriter içermektedir.



Şekil 4.1. Kural çıkarım tekniklerinin sınıflandırılması.

İfade Gücü: Çıkarılan kurallar aşağıdaki gibi çeşitli formda gösterilir.

- *Klasik sembolik kurallar* (ikili, önermeli) "EĞER …… O HALDE" ve "N içinde M" gibi. Önermeli mantık, mantıksal tanımların doğruluğu hakkında nedenleme için bir matematiksel modeldir. Tam ifadelerin sadece doğru değerleriyle ilgilenildiği ve nesneler arasındaki genel ilişki ve bağımlılıkları dikkate almadığı şeklinde sınırlamaları vardır.

- *Klasik olmayan kurallar:* Bulanık küme ve mantığa dayanan kurallar.

- *Birinci derece hüküm:* Hüküm mantığı, hükümler (belirli bir alandaki nesnelerin özelliklerini veya ilişkilerini tanımlayarak), niteliklendirmeler (keyfi alanlarda yer alan değişkenler) ve niteliklendirilmiş hükümler için çıkarım kuralları belirleyerek önermeli kuralları genişletir.

Kalite: Kalitenin ölçüleri şu şekilde özetlenebilir;

- *Kural doğruluğu:* Kurallar, eğitim sırasında görülmemiş, daha önceden görülmeyen örnekleri doğru bir şekilde sınıflandırmalıdır.

- *Kuralın uygunluğu:* Çıkarılan kuralların gizli bilgiyi tam gösterimiyle ilgilidir.

- *Kural uyumu:* YSA'nın, farklı eğitimlerde, görülmemiş örneklerin aynı sınıflandırmaları üretmesi şeklinde, kural kümeleri üretmesi halidir.

- *Kural anlaşılırlığı:* Kural sayısı ve eğitilen sinir ağından çıkarılan her kuralın ifade sayısıyla tanımlanır.

- *Kural ölçeklenirliği:* Kural çıkarım algoritmasının çalışma zamanının nasıl olduğu ve ağ, nitelik kümesi ve eğitim küme boyutu gibi faktörlerin fonksiyonu olarak değişen, çıkarılan modellerin anlaşılırlığının ölçüsüdür.

- *Kural genelliği:* Metodun özel eğitim rejimleri veya ağ mimarisinde sınırlamalar gerektirmesidir.

Yarı Şeffaflık: Çıkarılan kurallar ve temel yapay sinir ağının iç mimarisi arasındaki ilişkileri yansıtır. Analizci, eğitimci ve seçici olarak sınıflandırılan kural çıkarım tekniklerini sınıflandırır.

- *Analizci:* Kural çıkarımını ikili formda gizli ve çıktı birimler düzeyinde ele alır. Gizli katmanların aktivasyon değerlerini ve ağırlıklarını analiz ederek kurallar çıkarır.

- *Eğitimci:* Girdileri doğrudan çıktılara eşleştirmeye çalışır ve YSA'ları kara kutu olarak gösterir. Kuralları çıkarırken sadece girdi ve çıktı aktivasyonlarından faydalanır. Çıkarılan kural sayısı ve kuralların yapısı ele alınan sinir ağının yapısından ve ağırlık sayısından bağımsızdır.

- *Seçici:* Seçici yaklaşımlar hem analizci, hem de eğitimci tekniklerin elemanlarını içermektedir.

Algoritma Karmaşıklığı: Büyük veri kümeleri ve kural tabanları ile ilgili hesapsal sonuçlardır. Kural çıkarım tekniğini desteklerler.

Portatiflik: Ele alınan yapay sinir ağının özel eğitim yöntemlerini içine alabilme kabiliyetidir.

4.4. YSA'dan Kural Çıkarım Metotları

4.4.1. Analizci Metotlar

Analizci metotlarda, bir ağ her biri tek bir gizli katman nöronu içeren birçok farklı ağa ayrılır. Gizli katman nöronun aktivasyonu, bağlantılı girdi nöronların aktivasyon örneklerine dayanır ve ağırlıklar, ağın girdi katmanından bağlantıların eğilimine bakılarak elde edilir.

Analizci yaklaşımların temel özellikleri şunlardır;

- Bir sinir ağındaki her birimden kurallar veya ifadeler çıkarır ve bunları bütünleştirir.
- Eğitilen YSA'nın her bir biriminin çıktısı ikili birimlere denk gelir.

- Sinir ağındaki her bir birimi ele alır, böylece eğitim sonuçları birim vasıtasıyla anlaşılabilir.
- Genel olarak, kurallar ağdan arama ile çıkarılır.
- Birçok metot hesapsal karmaşıklıkta üsteldir.

Bu metotların olası dezavantajı, belirli bir mimariye özel olmalarıdır. Analizci metotlara örnek olarak aşağıdaki algoritmalar verilebilir.

- COMBO
- KT
- Tsukimoto'nun metodu
- FERNN
- REFANN
- RX
- Subset
- N içinde M
- RULEX
- RuleNet
- Partial-RE
- Full-RE
- BAB-BB

4.4.2. Eğitimci Metotlar

Eğitimci metotlar, ağın içyapısı veya ağın nasıl eğitildiği hakkında herhangi bir ön bilgiye ihtiyaç duymadan bir ağdan kurallar çıkarırlar. Ağı parçalamaya çalışmazlar. Bunun yerine, girdi nöronların kümesine dayanan olası tüm kurallardan oluşan kural alanı üzerinde çalışırlar.

Eğitimci metotların temel özellikleri şöyledir;

- YSA'yı kara kutu gibi gösterirler.
- Eğitilmiş sinir ağından örnekler üretir ve örneklerden kurallar meydana getirirler.

- İleri beslemeli ağlardan sadece girdi-çıktı eşleştirme yapısını dikkate alarak kurallar çıkarırlar.
- Kurallar sinir ağının tipi veya yapısı dikkate alınmadan çıkarılır.
- Bütün sinir ağının sonuçlarını sunarlar.
- Eğitim algoritmalarına dayanmazlar.

Eğitimci metotlara örnek olarak aşağıdaki teknikler verilebilir;

- Geçerlik Aralığı Analizi
- TREPAN ve
- BRAINNE
- BIO-RE

4.4.3. Seçici Metotlar

Seçici metotlar, yapay sinir ağlarını tek birim düzeyinde analiz eden fakat kuralları küresel düzeyde çıkaran, özellikle melez teknikleri birleştirmek için geliştirilmişlerdir.

Seçici metotların temel özellikleri;

- Analizci ve eğitimci tekniklerinin elemanlarını birleştirirler.
- YSA'lardan kurallar çıkarmakta sembolik öğrenme algoritmasını tamamlamak için eğitilmiş yapay sinir ağlarının iç mimarisi ve/veya ağırlık vektörü hakkında bilgi kullanırlar.

Seçici metotlara örnek olarak aşağıdaki teknikler verilebilir;

- REAL
- DEDEC
- GYAN

4.5. METASEZGİSELLERLE YAPAY SİNİR AĞLARINDAN KURAL ÇIKARIMI

Literatürde metasezgisel kullanarak yapay sinir ağlarından kural çıkarımına yönelik pek çok çalışma mevcuttur. Yapılan çalışmalarda genel olarak evrimsel algoritmalar kullanılmıştır. Sürü zekası teknikleri de evrimsel algoritmalar kadar yaygın olmamakla

birlikte, yapay sinir ağlarından kural çıkarımında kullanılmaktadır. Literatürde metasezgisellerle YSA'dan kural çıkarımına yönelik çalışmalar kısaca izleyen bölümde açıklanmıştır.

Evrimsel Algoritmalar: Zhang ve ark. [137], genetik algoritma kullanılarak budanan sinir ağından kurallar çıkarmak için RulExt algoritmasını geliştirmişlerdir. RulExt algoritmasının temelini standart çok katmanlı ileri beslemeli sinir ağı oluşturmaktadır. Aktivasyon fonksiyonu olarak sigmoid fonksiyonu kullanmışlardır. İleri beslemeli sinir ağını standart geri yayılım algoritması ile eğitmiş ve eğitilmiş ağı GA kullanarak basitleştirmişlerdir. Algoritmalarında GA'daki her bir popülasyon eğitilmiş ağın bağlantılarını ifade etmektedir. Bireyler, her bir elemanın bir bağlantıyı gösterdiği ikili kodlamalardır. İkili gösterimde "1" bağlantının varlığını, "0" ise bağlantı olmadığını ifade etmektedir. Girdi ve/veya çıktı bağlantılarının hepsinde bağlantı olmaması durumunda ilgili birimi iptal etmişlerdir. Amaçları doğruluğu koruyarak, olabildiğince bağlantı ve birimi budamaktır. Tahminleyici doğruluğu koruyarak, önemsiz bağlantıları budamış ve ilgisiz nitelikleri indirgemişlerdir. Algoritmaları ağ ağırlıklarını dikkate almamakta, aktivasyon fonksiyonları üzerine çalışmaktadır. Aktivasyon değerleri sürekli değerler olduğu için öncelikle bu değerleri kümeleme algoritması ile kesiklendirmişlerdir. Kesiklendirme işleminden sonra, M çıktı birimleri sayısını göstermek ve problemin sınıf sayısına eşit olmak üzere, budanan ağı M ağaca çevirmişlerdir. Bu yolla, her bir çıktı birimi bir ağaca dönüştürülmüştür. Her bir ağacın kökü bir çıktı birimini ve bunların dalları da ağdaki ilgili çıktı birimlerine girdi olan birimleri ifade etmektedir. Ağaçta dallanma girdi birimlerine ulaşılana kadar devam etmektedir. Bu algoritma ile yapraklar girdi niteliklerini, kök ise verinin sınıfını göstermektedir. Oluşturulan ağacı kökten yapraklara adım adım analiz ederek her bir sınıfla ilgili kuralları elde etmişlerdir. RulExt algoritmasında her bir sınıf bir kurallar kümesi oluşturmaktadır. Çıkarılan kurallar ağların doğruluğunu korumakla birlikte yorumlanabilir niteliktedir. Tez çalışmasında geliştirilen TAKKOYSA-sınıflandırıcısının aktivasyon fonksiyonu yerine ağ ağırlıklarına dayanarak kurallar çıkarması ve kural çıkarırken ağ üzerinde indirgeme yapmayarak, bütün ağı dikkate alması RulExt algoritmasından temel farklılıklarıdır.

Arbatlı ve Akın [138], YSA topolojisini optimize etmek için GA'ları kullanmışlardır. GA'yı uygulamaya rastgele topolojilerle, yani rastgele üretilen genotiplerle

başlamışlardır. Daha sonra her bir genotipi, fenotipe çevirmiş ve sinir ağı eğitim algoritması kullanarak eğitmişlerdir. Seçim, çaprazlama ve mutasyon operatörlerini uygulayarak yeni nesli elde etmişlerdir. Bu süreci önceden belirlenen nesil sayısına ulaşana kadar tekrarlamışlardır. Son iterasyondaki en iyi uyuma (en az hataya) sahip topolojiyi seçmişlerdir. Algoritmalarında YSA'ların genetik gösteriminde direk kodlama kullanmışlardır. Böylece fenotip, genotip tarafından tam olarak tanımlanabilmiştir. Genotipi ise, "1" bağlantının varlığını, "0" yokluğunu ifade etmek üzere ikili bir diziyle göstermişlerdir. Algoritmalarında başlangıç popülasyonunu rastgele üretilen genotiplerden oluşturmuş ve dizi uzunluğunu gizli birim sayısının bir fonksiyonu olarak ele almışlardır. Ağ topolojilerinin eğitilmesinde, geri yayılım algoritması ve birleşik gradyan öğrenme melezini, öğrenme algoritması olarak ele almışlardır. Uygunluk fonksiyonu olarak ise, ortalama kareli test hatasının tersini kullanmışlardır. Daha sonra eğitilen ve optimize edilen sinir ağlarından ayıran normal formda kurallar çıkarmak için analizci yaklaşımını kullanmışladır. Yaklaşımlarının altındaki temel fikir, YSA topolojisini optimize etmek için GA'yı kullanmak ve daha sonra optimize edilen topolojiyi kullanarak, girdi alanından ilgili girdileri çıkarmak ve son olarak bu ilgili girdileri kullanarak birleşik kurallar oluşturmaktır. Yaklaşımlarını deneysel olarak performans, doğruluk ve anlaşılırlık yönünden incelemişlerdir. Algoritmalarının tez çalışmasında geliştirilen TAKKOYSA-sınıflandırıcısından temel farklılığı, YSA topolojisini optimize ederek kural çıkarmalarıdır. Kural çıkarımında bir metasezgisel yerine KT gibi klasik bir kural çıkarım algoritması kullanmaları da algoritmalarının diğer bir farklılığıdır.

Fukumi ve Akamatsu [139], EA kullanılarak eğitilen sinir ağlarından kurallar çıkarmak için bir metot geliştirmişlerdir. Kullandıkları EA, deterministik mutasyonlu bir genetik algoritma ve deterministik mutasyonlu bir rastgele optimizasyon metoduna dayanmaktadır. Deterministik mutasyonu sinir ağ öğrenme sonuçları temeline uygulamışlardır. Bu, deterministik olarak uygunluk fonksiyonlarını artırmak için bireylerin kromozomlarını içermektedir. Basit kurallar çıkarabilmek için ağ boyutun evrimsel algoritmalar kullanılarak azaltılması gerekmektedir. Dolayısıyla EA'yı sinir ağ bağlantılarının sayısını azaltmak için kullanmışlardır. Algoritmalarında eğitimden sonra hayatta kalan ağ bağlantıları, örnek sınıflandırması oluşturmak için kuralları temsil etmektedir. Ayrıca, gizli katmanlarda ikili çıktılar üretmek için sigmoid fonksiyonlu

gizli birimler kullanmışlardır. Böylece, eğitilmiş sinir ağlarından basit kurallar çıkarılmasını sağlamışlardır. Simülasyon sonuçları, bu metodun basit bir ağ yapısı üretebileceği ve sonuç olarak Iris veri sınıflandırması için basit kurallar çıkarabileceğini göstermiştir. Tez çalışmasında geliştirilen TAKKOYSA-sınıflandırıcısı ile algoritmalarının en belirgin benzerlikleri sigmoid aktivasyon fonksiyonu kullanmaları, en belirgin farklılıkları ise metasezgisel algoritmaları kullanmalarındaki amaçtır. Fukumi ve Akamatsu GA'yı ağı indirgemekte kullanırken, TAKKOYSA-sınıflandırıcısı kural üretiminde kullanmaktadır.

Fukumi ve ark. [140], sinir ağlarından kural çıkarımına yeni bir metot önermişlerdir. Bir sinir ağını, eğitim kümesindeki düzeni göstermek için virüs enfeksiyonlu ve deterministik mutasyonlu genetik algoritma kullanarak oluşturmuşlardır. Basit kurallar çıkarmak için ağ boyutunun GA kullanılarak indirgenmesi gerekir. Bunu sağlamak için, modül yapılı bir GA kullanmışlardır. Her bir modülde, sigmoid ve yüksek düzey sinir ağları gibi farklı bir sinir ağ yapısını GA kullanarak oluşturmuşlardır. GA'da kromozomlar arasındaki ağ bilgileri virüs enfeksiyonu vasıtasıyla diğer modüllerle haberleşmektedir. Ağın tanımlama doğruluğunun belirlenmesinde, gizli katmanlarında ikili çıktılar üretebilmek için sigmoid fonksiyonlu gizli birimlerden faydalanmışlardır. Eğitimde, sigmoid fonksiyonlar gizli katmanlarda kullanılmıştır. Bu, eğitilmiş sinir ağlarından basit kurallar çıkarmalarına olanak sağlamıştır. Modüller üzerinde çalışması ve virüs enfeksiyonlu genetik algoritma kullanması Fukumi ve Akamatsu'nun [139] algoritmasından temel farklılıklarıdır. Bilgisayar simülasyonları yaklaşımın açık ağ yapıları ürettiğini ve sonuç olarak basit kurallar çıkardığını göstermektedir. Ancak YSA'da bağlantı sayısının çok olması durumunda algoritmaları iyi çalışmamakta ve uzun zaman almaktadır. Algoritmalarının modüller üzerinde çalışması ve veri gösterimi, TAKKOYSA-sınıflandırıcısından temel farklılıklarıdır.

Santos ve ark. [141], sinir ağlarından doğru ve anlaşılır kurallar çıkarmak için bir metot önermişlerdir. Yöntemleri ağ topolojisi oluşturma ve sinir ağlarından kurallar çıkarma alanlarını birleştirmektedir. Sundukları metot, iyi bir tek gizli katmanlı ileri beslemeli sinir ağı topolojisi bulmak için GA'ları kullanmaktadır. GA'da popülasyondaki her bir birey RPROP algoritması ile eğitilen bir sinir ağı topolojisini ifade etmektedir. Bu topoloji, daha sonra çıktısı bir sınıflandırma kümesi olan kural çıkarım algoritmasına geçmektedir. Kural çıkarımında basitliği nedeniyle RX algoritmasını kullanmışlardır.

Çıkarılan kuralların kalitesini tahminleyici doğruluk ve anlaşılırlıklarına göre değerlendirmişlerdir. Bu değerlendirmenin sonucunu ise bireyin uygunluk değeri olarak ele almışlardır. Uygunluk değeri tek bir kuralın kalitesi yerine bir kural kümesinin kalitesini değerlendirmektedir. Bu da yaklaşımlarının önemli bir avantajıdır, çünkü bu sayede kural etkileşimleri ortadan kalkmaktadır. Sonuç olarak, YSA topolojisinin kalitesi, çıkarılan kuralların tahminleyici doğruluk ve anlaşılırlıklarına göre değerlendirilmektedir. Hesaplanan kalite yeniden, aday yeni ağ topolojileri üretmek için kaliteyi dikkate alan GA'ya beslenmektedir. Sundukları sistemi üç yerel-alan veri kümesinde değerlendirmişlerdir ve yaklaşımın doğru olduğu sonucuna varmışlardır. Ağ topolojisini optimize ederken eş zamanlı olarak sınıflandırma kuralları çıkarmaları tezde geliştirilen algoritmadan temel farklılıklarıdır. Ayrıca TAKKOYSA-sınıflandırıcısından farklı olarak tek gizli katmanı dikkate almış, kural çıkarımından klasik metotlardan biri olan RX algoritmasını kullanmışlardır. TAKKOYSA-sınıflandırıcısında ise gizli katman sayısı bir veya iki katman şeklinde veri kümesine göre deney tasarımı ile belirlenmiştir.

Hruschka ve Ebecken [142, 143], sınıflandırma problemlerine uygulanan danışmanlı sinir ağlarından bilgi kazanımı konusunda bir çalışma yapmışlardır. Metodolojileri, gizli birim aktivasyon değerlerinin kümelenmesine dayanmaktadır. Sinir ağlarından kural çıkarımı için bir kümeleme genetik algoritması geliştirmişlerdir. Standart genetik operatörlerin kullanılmasına izin veren, sabit uzunluk yapısında kromozomlar üreten basit bir kodlama yapısı kullanmışlardır. Ayrıca bu tip gösterimlerin bazı zorluklarının üstesinden gelmek için, uygun bir algoritma geliştirmişlerdir. Buna ek olarak, başlangıç popülasyonunu oluşturmak için basit bir sezgisel kullanmışlardır. Bireylerin uyumunu, nesneler arasındaki öklit uzaklığa ve aynı zamanda her kümeye ait nesne sayısına göre belirlemişlerdir. İlk olarak bir sinir ağı paket programı kullanarak YSA'yı eğitip, girdi katmanından gizli katmana ve gizli katmandan çıktı katmanına ağırlıkları elde etmiş ve bu ağırlıklara göre girdi birimlerinden gizli birimlere aktivasyon denklemlerini oluşturmuşlardır. Kümeleme genetik algoritmasını kullanarak kümeleri ve bu kümelere düşen elemanları bulmuşlardır. Bu elemanların maksimum ve minimum değerlerini dikkate alarak, kümeleme genetik algoritmasından alınan sonuçlara göre aktivasyon denklemini yeniden yazmış ve kuralları elde etmişlerdir. Geliştirdikleri algoritmayı iki veri madenciliği referans problemine uygulamışlardır. Doğruluk açısından geliştirdikleri algoritma geliştirilmiş RX algoritmasına göre daha iyi sonuçlar vermektedir.

Algoritmalarının ağırlıklar yerine aktivasyon değerlerine bağlı kurallar üretmesi tez çalışmasında geliştirilen algoritmadan temel farklılığıdır.

Fu ve Wang [144], GA ve radyal taban fonksiyonlu sinir ağ sınıflandırıcısı kullanarak kurallar çıkarmak için yeni bir algoritma önermişlerdir. Her kuralın durum parçasındaki her girdi için aralık, genetik algoritmalar kullanılarak belirlenmiştir. Bir kromozomun uygunluğu ise çıkarılan kuralın doğruluğu vasıtasıyla belirlenmiştir. Çıkarılan kuralların karar sınırları hiper-dörtgen formundadır. Radyan taban fonksiyonlu sinir ağının eğitimi sırasında, sınıflandırma doğruluğunu korurken, gizli birim sayısını azaltmak için aynı sınıfla ilgili kümeler arasındaki büyük örtüşmelere izin verilmekte, daha sonra gizli birimleri çıktı birimlerine bağlayan ağırlıklar budanmaktadır. Simülasyonları yaklaşımlarının doğru ve anlaşılır kurallar çıkardığını göstermiştir. Algoritmalarının radyal taban fonksiyonlu sinir ağına dayanması, çok katmanlı algılayıcı modeline dayanan ve TAKKOYSA-sınıflandırıcısından temel farklılığıdır.

Fukumi vd., [145], yapay sinir ağlarından kural çıkarımına yeni bir yaklaşım sunmuşlardır. "Eğer….o halde" formunda bir kuralı, eğitim verisinin özelliklerini dikkate alarak üretmiş ve daha sonra bunları ağırlık değerleri olarak YSA'ya eklemişlerdir. Bu sinir ağını, eğitim verisini kullanarak eklenen kuralları indirgemek için eğitmişlerdir. Eğitim sırasında küçük boyutlu sinir ağ sistemleri üretebilmek için yapı öğrenme algoritmasını kullanmışlardır. YSA eğitiminden sonra, kuralları bu sinir ağ sisteminden çıkarmışlardır. Ayrıca ikili çıktılar üretmek için YSA gizli birimlerinde sigmoid fonksiyon kullanmışlardır. Böylece eğitilmiş YSA'dan basit kurallar çıkarabilmişlerdir. Simülasyonları ve klasik kural çıkarım metotları ile karşılaştırmaları yaklaşımlarının iyi bir ağ yapısı oluşturduğunu ve basit kurallar elde edebildiğini göstermektedir. Daha önce geliştirdikleri algoritmalardan [39, 140] farklı olarak bu çalışmalarında kuralları eğitimden önce YSA'ya beslemişlerdir. Bu algoritmalarının TAKKOYSA-sınıflandırıcısından da temel farklılığıdır. TAKKOYSA-sınıflandırıcısı öncelikle YSA'yı eğiterek ağırlıkları elde etmekte daha sonra ise bu ağırlıklara göre kurallar üretmektedir.

Dorado ve ark. [146], kural araştırma tekniği olarak GP'yı kullanan ve her türdeki YSA'na uygulanabilen bir kural çıkarım sistemi önermişlerdir. Sistemleri her türdeki yapay sinir ağlarına uygulanabilmektedir. Geliştirdikleri sistemin hiçbir yapay sinir ağı

mimari ve eğitim ihtiyacı yoktur, doğrudur ve tanımlama yeteneği oldukça yüksektir. Deneysel çalışmaları yöntemlerinin doğru sonuçlar verdiğini ve her türdeki yapay sinir ağlarına uygulanabildiğini göstermektedir. Algoritmaları YSA'ları kara kutu olarak ele alan eğitimci yaklaşıma girmektedir. Kural gösteriminde ise tezde geliştirilen TAKKOYSA-sınıflandırıcısından farklı olarak karar ağaçlarından faydalanmışlardır.

Tsukimoto ve Hatano [147], eğitilmiş üç katmanlı sinir ağı birimlerinden ikili fonksiyonlar çıkarmak için bir algoritma geliştirmişlerdir. Çıkarılan ikili fonksiyonlar gizli birimleri anlaşılır hale getirmektedir. Bir gizli birimden çıkarılan ikili fonksiyon basit olduğunda, gizli birim fonksiyonel olarak belirlenmiş demektir. Fakat ikili fonksiyon karmaşıksa, bu durumda birim fonksiyonel olarak belirgindir şeklinde yorumlanamaz. Bu nedenle çalışmalarında, gizli birimlerden çıkarılan ikili fonksiyonların basitliği sayesinde, gizli birimlerin fonksiyonel belirginliğini ele almışlardır. İkili fonksiyonların basitliğinin değerlendirilmesi ile ilgili çeşitli metotlar mevcuttur. Tsukimoto ve Hatano, ikili fonksiyonların basitliğini ikili fonksiyonun derece ve ifade sayısına göre tahmin etmişlerdir. Algoritmalarında, bir gizli birim birkaç ifade ile ikili fonksiyona iyi tahminlendiğinde, fonksiyonel olarak belirgin demektir. Bir gizli birimin fonksiyonel belirlenmesi, gizli birim ile gizli birimden çıkarılan ikili fonksiyon arasındaki hata vasıtasıyla değerlendirilmektedir. Dolayısıyla, fonksiyonel belirginliğin değeri başlangıç ağırlık değerlerinin bir fonksiyonudur ve bu nedenle başlangıç ağırlık değerlerini genetik algoritmaların bireyleri olarak kodlamışlardır. Uygunluk fonksiyonunu ise gizli birimler ve gizli birimlerden çıkarılan ikili fonksiyonlar arasındaki hataya göre tanımlamışlardır. Optimizasyonu genetik algoritma kullanarak gerçekleştirmişlerdir. Algoritmalarında ilk olarak GA popülasyondaki sinir ağlarının başlangıç ağırlık değerlerini belirlemekte kullanılmış, daha sonra sinir ağı veriyi geri yayılım metodu kullanarak öğrenmiş ve son olarak sinir ağları fonksiyonel belirginlik için değerlere göre yaşama şansı elde etmişlerdir. Sonuç olarak, geri yayılım algoritmasını veriyi öğrenmekte, GA'yı ise fonksiyonel belirginlikte kullanmışlardır. Algoritmalarının ikili fonksiyonlar üretmesi, sınıflandırma kuralları sunan TAKKOYSA-sınıflandırıcıdan temel farklılığıdır.

Markowska-Kaczmar ve Lipinski [148], "GenPar" ismini verdikleri, sinir ağlarından kural çıkarımı algoritmasını geliştirmişlerdir. Algoritmaları pareto optimizasyonlu genetik yaklaşıma dayanmaktadır. Çıkardıkları kurallar "eğer… o halde" formunda

önermeli kurallardır. Uygunluk fonksiyonunda doğruluk ve anlaşılırlık ölçütlerini ele almışlardır. GenPar, sınıflandırma problemlerinde sinir ağlarının GA'larla tanımlanması için bir metottur. Yöntemlerini bilinen test problemleri üzerinde test etmiş ve sonuçlarını tartışmışlardır. Yaptıkları deneysel çalışmalar GenPar metodunun kesikli ve sürekli nitelikler için etkin bir şekilde çalıştığını göstermektedir. Geliştirdikleri algoritma veri kodlama, uygunluk değerlendirme gibi birçok yönden tez çalışmasında geliştirilen algoritmadan ayrılmaktadır.

Elalfi ve ark. [149], eğitilmiş YSA aracılığıyla veritabanlarından doğru ve anlaşılır kurallar çıkarmak için GA kullanarak yeni bir yaklaşım sunmuşlardır. Yeni algoritmaları YSA eğitim algoritmalarına dayanmamakta ve eğitim sonuçlarını değiştirmemektedir. GA'yı, çıktı düğümü k'nın çıktı fonksiyonu ψ_k'yı maksimize eden girdi niteliklerinin optimal değerlerini bulmak için kullanılmışlardır. Optimal kromozomu kodlamış ve ilgili sınıfa ait kuralı elde etmekte kullanmışlardır. Algoritmaları kesikli ve sürekli değişkenlere uygulanabilmekte, gizli birim aktivasyon fonksiyonu üzerine herhangi bir tahmin yapmamaktadır. Buna ek olarak eğitilmiş sinir ağlarında herhangi bir sayıda gizli katmanı ele alabilmektedir. Veri kümesini ikili formda kodlamış ve eğitilmiş yapay sinir ağlarından elde edilen ağırlıkları kullanarak YSA çıktı fonksiyonunu maksimize eden girdi vektörünü genetik algoritma kullanarak bulmuşlardır. Algoritmaları her sınıf için ayrı çalışmaktadır. Elde edilen kuralları YSA çıktı fonksiyon değerlerine göre büyükten küçüğe doğru sıralamışlardır ve bu sıradan belirli bir düzeye kadar olan kuralları çıkarılan kurallar olarak saklamışlardır. Daha sonra her bir kuralı basitleştirerek dilsel kurallara dönüştürmüşlerdir. Tez çalışması sırasında geliştirilen algoritma Elalfi ve arkadaşlarının veri gösterimini kullanmaktadır. TAKKOYSA-sınıflandırıcısında, Elalfi ve arkadaşlarının algoritmasından farklı olarak kurallar YSA çıktı fonksiyon değerlerine göre sıralanıp belirli kurallar saklanmamakta, ele alınan uygunluk fonksiyonuna göre YSA çıktı fonksiyonu yüksek olan kurallar arasından kural indirgeme ile kurallar seçilmektedir. Kural üretiminde GA yerine KKO kullanılması da algoritmalar arasındaki diğer bir farktır.

Markowska-Kaczmar [150], [148]'deki çalışmalarından farklı olarak sinir ağlarından kural çıkarmak için genetik algoritmalara dayanan GEX (Genetic rule EXtraction) algoritmasını geliştirmiştir ve metodun parametrelerinin son sonuçlardaki etkilerini

deneysel çalışmalarla incelemiştir. Metodu, her sınıfın ayrı popülasyonları içerdiği evrimsel yaklaşıma dayanmaktadır. İlk olarak genetik parametrelerin etkilerini incelemiş, daha sonra ise kural çıkarımının etkinliğini etkileyen parametreleri test etmiştir. İkili, sürekli ve kesikli değişkenler için farklı kromozom yapıları kullanmıştır. GEX algoritmasında her popülasyonu, bir sınıfın kurallarını içerecek şekilde özelleştirmiştir. Bu da sınıflandırma probleminde mevcut olan sınıf sayısı kadar popülasyon olması anlamına gelmektedir. Popülasyondaki her bir birey bir kuralı ifade etmektedir. Uygunluk fonksiyonu olarak doğruluk, eksiklik, tamlık ve anlaşılırlık ölçütlerinin bir fonksiyonunu kullanmıştır. Deneysel çalışmaları farklı tipteki örnekler için GEX algoritmasının kural çıkarmakta esnek bir metot olduğunu göstermiştir. Kullandığı uygunluk fonksiyonu, metasezgisel algoritma ve algoritmasında ele aldığı amaç (parametrelerin etkisini incelemek) tezde geliştirilen algoritmadan ayrılan yönleridir.

Tokinaga ve ark. [151], zeki ve açıklayıcı değerlendirme sistemi kurmak için genetik programlanmaya dayanan sinir ağı kural çıkarma tekniklerinin kullanımını ele almışlardır. Sadece GP'ye dayanan sınıflandırma kural çıkarımı karar verme problemlerine doğrudan uygulanabilir. Fakat orijinal GP metodunun sınıflandırma kuralları aritmetik tanımlamalar, temel mantık ifadeleri ile açıklanan birçok ifade içerir ve bu da kural üretim sürecini çok karmaşık hale getirir. Bu nedenle, gerçek uygulamalarda GP prosedürünün doğrudan uygulamasından kaçınıp aritmetik tanımlamalarla basit ve ilgili sınıflandırma kuralları elde etmek için sinir ağları ve ikili sınıflandırmadan faydalanmışlardır. Basitleştirilmiş kuralları azaltmak için eğitilmiş sinir ağlarının girdi değişkenlerinin kesiklendirme ve ayrıştırma kabiliyetine odaklanmışlardır. Girdi ve çıktı birimlerini birleştirme süreci GP kullanılarak gerçekleştirilmiştir. Girdi değişkenleri kullanarak sinir ağlarının eğitiminden sonra, nöronlar arasındaki ağırlıklara basit fakat sağlam ikili ifadeler elde etmek için budama süreci uygulamışlardır, bunlar ifade olarak kullanılan sınıflandırma kurallarıdır. Daha sonra GP, en son kuralı elde etmek için uygulanmıştır. Uygulamaları neticesinde sundukları kural üretim yönteminin performansının karşılaştırılabilir olduğunu ve iyi sonuçlar verdiğini görmüşlerdir. Tokinaga ve arkadaşlarının algoritması, tez çalışmasında geliştirilen algoritma gibi analizci bir metottur. Ancak Tokinaga ve arkadaşları ağ yapısını budamakta ve ağırlıklar üzerinden kural çıkaran TAKKOYSA-

sınıflandırıcısının aksine aktivasyon değerlerinden faydalanarak kurallar çıkarmaktadırlar.

Markowska-Kaczmar ve Trelak [152], daha önce yaptıkları çalışmalardan [148,150] farklı olarak, sinir ağlarından bulanık kural çıkarımında REX metodunu sunmuşlardır. Yöntemleri evrimsel algoritmalara dayanmaktadır. Evrimsel algoritmaların arama sürecinde, YSA'ların performansını tanımlayan bir kurallar kümesi bulmuşlardır. Evrimsel algoritma ayrıca, doğru bulanık kümelerin elde edilmesinden sorumludur. REX Pitt ve REX Michigan olmak üzere iki yaklaşımı karşılaştırmışlardır. Bunlar arasındaki temel farklılık bir kromozomda bulunan bilgide yatmaktadır. REX Pitt'de bir birey kurallar kümesini gösterirken REX Michigan'da bir kuralı göstermektedir. Elde edilen sonuçları bilinen yöntemlerle karşılaştırmışlardır. REX Pitt, küçük sayıda bulanık kurallar üreten çok iyi etkinliğe sahipken, REX Michigan daha düşük kaliteli kurallar üretmektedir. Algoritmalarının bulanık kural çıkarımı üzerine çalışması, tezde geliştirilen algoritmadan temel farklılığıdır.

Hruschka ve Ebecken [7], çok katmanlı algılayıcılardan kurallar çıkarmak için kümeleme genetik algoritmasını kullanmışlardır. Sundukları metot, gizli birim aktivasyon değerlerine dayanmakta ve iki temel adımdan oluşmaktadır. İlk adımda gizli birim aktivasyon değerlerinin kümelerini bulmak için kümeleme genetik algoritması uygulanmakta, daha sonra ikinci adımda girdilerle bağıntılı olarak, bu kümeleri tanımlayan sınıflandırma kuralları üretilmektedir. Yöntemleri, eğitilmiş sinir ağlarından kurallar çıkarmaya odaklanmakta ve gizli birim aktivasyon değerlerinin kümelenmesine dayanan RX algoritmasını temel almaktadır. Eğitilmiş sinir ağlarından kural çıkarmakla ilgili bazı metotlar sadece bağlantı ağırlıkları bilgisine ihtiyaç duyarken, Hruschka ve Ebecken'in metotları gibi metotlar aynı zamanda eğitim kümesine de ihtiyaç duymaktadır. Algoritmalarında başlangıçta tanımlanması gereken küme sayısıyla birlikte kümeleme sürecinin optimizasyonunu sağlamak için kümeleme problemlerine yönelik tasarlanan genetik algoritmalardan faydalanmışlardır. Böylece çıkarılan kuralların hem kalitesini, hem de sayısını optimize etmişlerdir. Yöntemleri "Eğer…. o halde……." formunda kurallar oluşturmakta ve herhangi bir özel çok katmanlı eğitim algoritması gerektirmemektedir. Algoritma, kesikli ve sürekli değişken içeren sınıflandırma problemlerine uygulanabilir. Kural çıkarım algoritmasının karmaşıklığı, uygulanan kümeleme genetik algoritmasına bağlıdır. Kümeleme genetik algoritması

basit bir kodlama şemasına dayanmaktadır. N örnekten oluşan bir veri kümesi için genotip (N+1) pozisyondan oluşan tamsayılı bir vektördür. Her pozisyon bir örneği ifade etmekte ve son gen küme sayısını temsil etmektedir. Sundukları yaklaşımı veri madenciliğinde referans olan dört veri kümesinde ve bir gerçek hayat veri kümesinde deneysel olarak değerlendirmişlerdir. Algoritmaları [142,143]'de geliştirdikleri algoritmaya dayanmaktadır. Önceki çalışmalarından farklı olarak algoritmalarının performansı değişik veri kümeleri üzerinde analiz etmiş ve geliştirmişlerdir. Algoritmalarının aktivasyon değerleriyle çalışması, aktivasyon değerlerini içeren kurallar oluşturması ve veri gösterimi TAKKOYSA-sınıflandırıcısı ile temel farklılıklarıdır.

Kahramanlı ve Allahverdi [153], eğitilmiş adaptif sinir ağlarından kurallar çıkarmak için yapay bağışıklık sistemi algoritması kullanan yeni bir yöntem sunmuşlardır. Çalışmalarının amacı, yeni bir adaptif aktivasyon fonksiyonu ve yapay bağışıklık sistemi kullanarak eğitilmiş sinir ağlarından kural çıkarımında bir metot geliştirmektir. Algoritmaları temelde Elalfi ve arkadaşlarının [149] çalışmasına dayanmaktadır. Yapay sinir ağlarını adaptif bir aktivasyon fonksiyonuyla eğitmiş ve üç parametreli bir nöron-adaptif aktivasyon fonksiyonu önermişlerdir. Algoritmalarını iki referans veri kümesinde değerlendirmiş ve elde ettikleri kuralları doğruluk açısından literatürdeki bazı algoritmalarla karşılaştırmışlardır. Ancak elde ettikleri kural sayıları çok yüksektir. Veri kodlama ve kural gösterimi, TAKKOYSA-sınıflandırıcısı ile algoritmaları arasındaki benzerliklerdir. Algoritmalar arasındaki temel farklılıklar olarak kullanılan aktivasyon fonksiyonları, kural üretiminde kullanılan algoritma ve kural indirgeme yöntemi sıralanabilir.

Sürü Zekası Teknikleri: Yapay sinir ağlarından kural çıkarımında kullanılan diğer metasezgisel teknikler parça sürü optimizasyonu ve karınca koloni optimizasyonu algoritmalarıdır. Parça sürü optimizasyonu ile ilgili çalışmalarda daha çok bulanık yapay sinir ağlarından kurallar çıkarımı ele alınmıştır. Karınca koloni optimizasyonu ile ilgili yapılan az sayıda çalışma mevcuttur ve bu çalışmalar da eğitilmiş sinir ağlarından kural çıkarımına yönelik değildir. Bu algoritmalar ile yapılan çalışmalar şöyle özetlenebilir;

Zhenya vd. [154], parça sürü optimizasyonuyla bulanık sinir ağlarından kurallar çıkarmakta dört katmanlı bir adaptif bulanık sinir ağını girdi çıktı örneklerinden bilgi kazancı elde etmek için sunmuşlardır. Girdi parametrelerinin gerekli üyelik fonksiyonlarını içeren ağ parametrelerini ve izleyen parametreleri değiştirilmiş parça sürü algoritması kullanarak belirleyip, ayarlamışlardır. Ayrıca bulanık sinir ağını eğitmekte geliştirilmiş PSO'dan faydalanmışlardır. Eğitilen ağı daha sonra budamışlar, böylece genel kurallar çıkarmış ve açıklamışlardır. Deneysel sonuçların diğer bulanık sinir ağı yaklaşımlarıyla karşılaştırılarak elde edilebileceğini vurgulamışlardır. Geliştirdikleri algoritma bulanık sinir ağları üzerinde çalıştığı için tez çalışmasında geliştirilen algoritma ile tamamen farklı yapıdadır.

Ma vd. [155], bulanık sinir ağını optimize etmek için parça sürü optimizasyonu algoritmasına dayanan bir budama algoritması geliştirmişlerdir. Geliştirdikleri algoritma hem bulanık sinir ağının topolojisini, hem de ağırlık parametrelerini içermektedir. Kullandıkları ağ yapısı, bulanık ağırlıklandırılmış nedenleme metoduna dayanan adaptif bir ağdır. İyileştirici bulanık ağırlıklandırılmış nedenleme metodunda, olası bütün bulanık kurallar listelenmeye çalışılır. Ağırlıklandırılmış bulanık sinir ağı, yedi katmanlı geri beslemeli bir ağdır. Algoritma, ihtiyaçlara göre bulanık sinir ağının yaklaşık en iyi yapısını otomatik olarak elde edebilir. Budama algoritmasında bulanık kurallar, gereksiz bağlantılar kaldırılarak elde edilebilir. Sonuçta uygun ağ ağırlıkları ve bulanık kurallar elde edilir. Ancak PSO, doğrudan ağırlıklandırılmış bulanık sinir ağına uygulanamaz. Standart PSO ile karşılaştırıldığında, üyelik fonksiyonunun optimizasyonunu ve ağırlıklandırılmış parametrelerin optimizasyonunu kendi problemlerine göre güncellemişlerdir. Deneysel çalışmaları, algoritmalarının kabul edilebilir ve etkin olduğunu göstermektedir. Algoritmalarının, TAKKOYSA-sınıflandırıcısından ayrılan en belirgin özellikleri bulanık sinir ağları üzerinde çalışıp, kümeleme ile ağı indirgeyerek kural çıkarması ve geri beslemeli YSA'lar üzerinde çalışmasıdır.

Chen ve ark [156], sinir ağı grubuna dayanan, Ant-Classifier isimli bir karınca koloni sınıflandırma kural çıkarımı tekniği geliştirmişlerdir. Ayrıca sinir ağı grubunu ve Ant-Classifier sistemini birleştiren NeAnt algoritmasını sunmuşlardır. NeAnt algoritmasında sinir ağı grubu, orijinal eğitim kümesinin ön işlenmesinde, Ant-Classifier algoritmasını ise yeni eğitim kümesinden kurallar çıkarmakta kullanılmıştır. Deneysel çalışmaları

NeAnt ve Ant-Classifier algoritmalarının her ikisinin de genelleme yeteneğinin çok iyi olduğunu göstermektedir. Geliştirdikleri algoritma eğitilmiş YSA'lardan kural çıkarımına odaklanmayıp, yapay sinir ağı grubunu sadece eğitim kümesinin oluşturulmasında kullandığından dolayı tez çalışmasında geliştirilen algoritma ile ortak noktaları bulunmamaktadır.

Sivagaminathan ve Ramakrishnan [157], nitelik seçimi için KKO ve yapay sinir ağlarına dayanan melez bir metot sunmuşlardır. Yapay sinir ağlarını sınıflandırma fonksiyonu olarak, karınca koloni optimizasyonunu ise değerlendirme algoritması olarak kullanmışlardır. Melez algoritmada her bir karıncanın görevi bir çözüm alt kümesi oluşturmaktır. Sundukları melez modeli tıbbi tanı alanındaki veri kümelerini kullanarak göstermişlerdir. Geliştirdikleri algoritma nitelik seçimi üzerine olduğu için tez çalışmasında geliştirilen sınıflandırma kural çıkarımı algoritması ile ortak noktaları yoktur.

4.6. Sonuç

Tez çalışmasında geliştirilen algoritmanın eğitilmiş yapay sinir ağlarından kural çıkarımı algoritması olması nedeniyle bu bölümde YSA'dan kural çıkarımı hakkında bilgiler verilmiştir. Kural çıkarımı algoritmaları sınıflandırılarak, her bir sınıfın taşıdığı özellikler açıklanmıştır.

Ayrıca bu bölümde literatürde yer alan YSA'dan evrimsel algoritmalar ve sürü zekası gibi metasezgisellerle kural çıkarım algoritmaları incelenerek özetlenmiş, tez çalışmasında geliştirilen TAKKOYSA-sınıflandırıcısı ile benzer ve farklı yönleri ortaya konmuştur.

5. BÖLÜM
SÜRÜ ZEKASI VE KARINCA KOLONİ OPTİMİZASYONU

5.1. Giriş

Bu bölümde eğitilmiş yapay sinir ağlarından sınıflandırma kuralları üretmekte kullanılan, zeki optimizasyon tekniklerinden karınca koloni optimizasyonu ele alınmıştır. İlk olarak sürü zekası kavramı kısaca açıklandıktan sonra, karınca kolonilerinin ve yapay karıncaların davranışlarına değinilmiştir. Daha sonra ise KKO algoritmaları sınıflandırılarak açıklanmıştır.

Tez çalışmasında geliştirilen algoritmanın temelini oluşturması nedeniyle, sürekli problemler için geliştirilen TAKKO algoritması detaylı olarak incelenmiştir. Son bölümde ise KKO algoritmalarının temel özellikleri özetlenmiştir.

5.2. Sürü Zekası

Sürü, birbirleriyle etkileşen dağınık yapılı ajanlar grubu olarak tanımlanır. Burada ajanlar balık, kuş, termit, arı, karınca vb. olabilir. *Zeka* ise, insanların zihinsel kabiliyetlerini tanımlayan bir terimdir. Bu iki terimin birleşmesinden meydana gelen *Sürü Zekası*, sosyal böcek kolonileri veya diğer canlı toplumlarının kolektif davranışlarından esinlenerek algoritmalar veya dağınık problem çözme planları tasarımına yönelik bir teşebbüsü kapsayan bir alan olarak tanımlanır [158]. Başka bir ifadeyle SZ, merkezi olmayan, kendi kendini yöneten sistemlerde toplu davranış çalışmasına dayanan bir yapay zeka tekniğidir. Sürü zekası, termit veya bal arılarının yuva yapımı, karınca kolonilerinin yiyecek arama davranışı, balıkların sürü halinde yüzmesi, kuşların sürü halinde uçması gibi doğadaki olaylardan esinlenilerek ortaya çıkmıştır.

SZ sistemleri genel olarak, birbirleriyle ve çevreleriyle etkileşen basit ajanlar topluluğundan oluşur. Birey olarak ajanların nasıl davranacağını dikte eden merkezileş-

miş bir kontrol yapısı yoktur. Bu ajanlar arasındaki yerel etkileşimler çoğunlukla küresel davranışın ortaya çıkışını gösterir. Bu tip sistemlere örnekler doğada mevcuttur. Hayvan kolonilerindeki yeteneklere örnek olarak böcekler verilebilir. Bir böcek sadece birkaç yüz beyin hücresine sahip olmasına rağmen, böcek davranışları mimari şaheserler ve ayrıntılı iletişim sistemleri oluşturabilecek yetenektedir [159].

Sürülerde bulunan özellikler olarak rastgelelik, toplu davranış, kendi kendine organizasyon ve etkileşim sıralanabilir. Bu özellikler aşağıda kısaca açıklanmaktadır;

Rastgelelik: Doğadaki adaptasyon, "rastgele" gerçekleşen durumları içeren bir süreçtir. Rastgelelik doğada ihtimale dayalı bir durumdur ve aynı duruma ait tekrar eden olaylar farklı sonuçlarla neticelendiğinde ortaya çıkar. Adaptif bir bilgisayar programında, rastgelelik belirsizlik ve yenilik olmak üzere iki kavramı ortaya çıkarır. Bir problem çözülürken nereden başlanacağı veya hangi yöne gidileceği bilinmemektedir, fakat bir yere veya bir yöne gidileceği kesindir. Bu durumda rastgele bir yön herhangi bir çözüm kadar iyidir. Ayrıca rastgelelik bazı durumlarda, yeni ve başarılı sonuçların ortaya çıkmasına neden olabilir.

Toplu davranış: Bir böcek kolonisi incelendiğinde böceklerin toplu hareket etmesine rağmen, aslında her bir basit böceğin kendi ajandası olduğu görülür. Bir koloni toplu halde çok organize çalışmaktadır. Burada organizasyonu sağlayan bir yöneticiye ihtiyaç yoktur, çünkü bu organizasyon görünmez bir etkileşimle sağlanır. Bu etkileşim dolaylı (etki ile) ve dolaysız (çevre ile) olmak üzere iki şekilde ortaya çıkar.

Kendi kendine organizasyon: Kendi kendine organizasyon, birçok farklı bireyden meydana gelen bir grubun, değişen şartlardan dolayı kendi durumunu değiştirerek yeni bir düzen kazanmasıdır. Ayrıca bir sistemin alt seviyedeki bileşenleri arasındaki etkileşimlerinden küresel seviyedeki yapıların ortaya çıkması anlamında kendi kendine organizasyon bir dinamik mekanizmalar bütünüdür. Karınca, arı, termit gibi böceklerin kendilerine göre basit, etkileşimli yaratıklar oldukları farz edilerek karmaşık kolektif davranışı açıklamanın mümkün olduğu düşünülmektedir. Arı, karınca, termit gibi kendi kendine organize olabilen böcekler hakkındaki bilgiler zeki sistem tasarımına uygulandığında esneklik, değişen çevreye uyum gibi çok güçlü etki ve araçlar ortaya çıkarır.

Etkileşim: Etkileşim (stigmergy) kelimesi ilk olarak, termitlerin yuvalarını yeniden inşa etme sürecindeki düzenlerini ve görev koordinasyonlarını açıklamak için kullanılmıştır [160]. Etkileşim bir böceğin hareketi başka bir böceğinin hareketinden etkilendiğinde ortaya çıkar. Etkileşim, bireysel davranışın çevreyi geliştirmesi sonucunda diğer bireylerin davranışlarının ve koloni seviyesindeki toplam davranışların gelişmesini açıklar. Burada çevre haberleşme ortamıdır. Eğer çevre, dış bir etki sonucu değişirse, bu değişim sanki koloni aktiviteleri sonucu oluşmuş gibi değerlendirilir. Böylece koloni, bireylerin aynı davranışı göstermesiyle kolektif bir cevap verebilir. Etkileşime örnek olarak, sosyal böceklerdeki dolaylı etkileşim verilebilir, çünkü kendi kendine organizasyon böceklerin kendi aralarında etkileşim kurmasını gerektirir. Dolaylı bir etkileşim, bir böceğin çevre şartlarını değişikliğe uğratması, diğerlerinin de bu yeni çevre şartlarına sonraki bir zamanda cevap vermesidir.

Sürü zekasının karınca koloni optimizasyonu, parça sürü optimizasyonu, arı koloni algoritması (AKA) ve stokastik difüzyon arama olmak üzere dört alt başlığı vardır. Bu teknikler gezgin satıcı problemi, karesel atama problemi, grafik boyama, küme bulma, iş çizelgeleme, arama motorları, iş yükü dengeleme gibi bir çok farklı alana uygulanmaktadır.

5.3. Karınca Koloni Optimizasyonu

Karınca kolonileri kör olmalarına rağmen, yuva ve yiyecek kaynağı arasındaki en kısa yolu bulma yeteneğine sahiptirler. Deneysel çalışmalar bu kabiliyetin karıncalar arasındaki kokuya dayalı kimyasal bir haberleşmenin sonucu olduğunu göstermiştir. Gerçek karıncaların yiyecek arama davranışlarından esinlenerek geliştirilmiş ve bir algoritmalar sınıfı olan karınca koloni optimizasyonu çok ajanlı sistemlerdir.

KKO algoritması ortaya atıldığı ilk günden itibaren bilim camiasının büyük ilgisini çekmiştir. Bu ilginin nedeni karınca koloni optimizasyonunun çok yönlü, sağlam ve popülasyon tabanlı bir algoritma olmasıdır. İlk karınca algoritması Dorigo ve öğrencileri [161] tarafından gezgin satıcı problemi ve karesel atama problemi gibi zor kombinatoryal optimizasyon problemleri için geliştirilmiştir. Daha sonra pek çok araştırmacı çeşitli kesikli optimizasyon problemlerine karınca-tabanlı algoritmaları uygulamışlardır.

Karınca optimizasyonunun temel özellikleri olarak şunlar sıralanabilir;

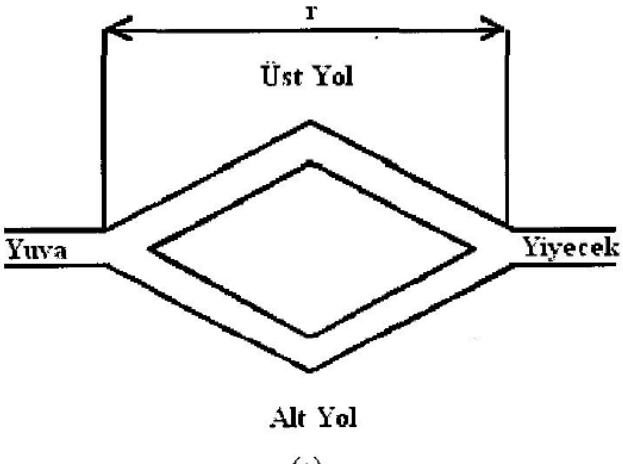
- Zeki davranış biçimi (koloni halinde en iyi çözümü bulabilme kabiliyeti zeka göstergesidir ve bu da yapay zeka grubuna girdiğini gösterir)
- Kombinatoryal optimizasyon (çok sayıda alternatif arasından en iyisini seçebilme yeteneği bu kombinatoryal optimizasyon problemlerinde iyi bir araç olmasını sağlar),
- Pozitif geri-besleme (yollara bırakılan feremon maddesi miktarı daha iyi çözümlerin bulunmasına yardımcı olur),
- Negatif geri-besleme (az kullanılan yolda feremon miktarının buharlaşma sonucu azalması ve sık kullanılan yolda buharlaşma olsa dahi feremon miktarının artması iyi çözümlerin bulunmasına yardımcı olur),
- Dağıtılmış hesaplama (erken yakınsamayı engeller),
- Olasılık özelliği (yol seçimi yolda depolanan feremon miktarına göre belli bir olasılıkla yapılır)
- Yapıcı aç-gözlü sezgiselin kullanımı (ilk aşamalarda karıncaların kabul edilebilir çözümleri bulmasında yardımcı olur).

5.3.1. Karıncaların Yiyecek Arama Davranışı

Karıncaların yiyecek kaynağı ile yuvaları arasındaki en kısa yolu herhangi bir görünür, merkezi veya aktif koordinasyon mekanizması olmadan bulması yiyecek aramada rastgele bir davranışın sonucudur [162, 163]. Çoğu karınca kolonilerinde, karıncalar ilk olarak yuvalarının çevresindeki alanı rastgele ararlar. Yiyecek kaynakları bulduklarında, yiyeceğin miktar ve kalitesini değerlendirdikten sonra yiyeceğin bir kısmını tekrar yuvalarına taşırlar. Bu geri dönüş sırasında karıncalar, diğer karıncaların aynı kaynağı bulabilmeleri için yiyeceğin kalite ve miktarına bağlı olarak yollarına feremon adı verilen kimyasal bir madde depolarlar. Yolda depolanan feremon miktarı diğer karıncaları belirli bir olasılıkla yiyecek kaynağına yönlendirir. Bu feremon miktarı ile karıncalar arasındaki dolaylı iletişim, karıncalara yiyecek kaynağı ile yuvaları arasındaki en kısa yolu bulma olanağı sağlar.

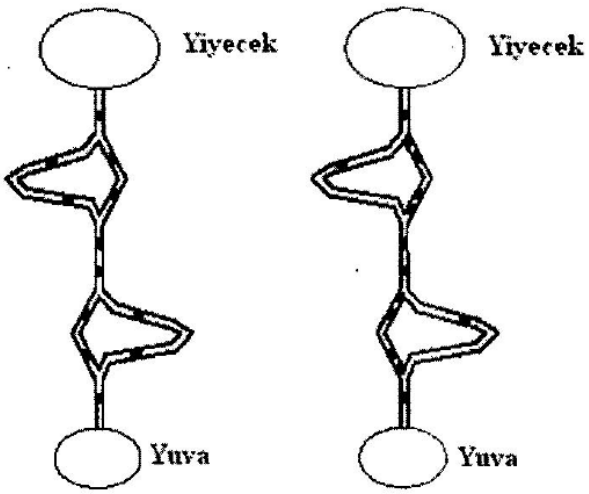
Karıncaların yiyecek arama davranışı ile ilgili ilk çalışmayı Arjantin karıncaları üzerinde Deneubourg vd. [164] yapmıştır. Şekil 5.1.'de gösterilen bu deneyde yuva

yiyecek kaynağından eşit uzunlukta iki dal içeren köprüyle ayrılmıştır. Başlangıçta iki dalda feremon içermemektedir. Belirli bir zaman geçtikten sonra, karıncaların çoğu tarafından bu dallardan biri seçilmiş ve iki dalın da eşit uzunlukta olmasına rağmen çoğu karınca bu dalı takip etmiştir. Buradaki bir yolda daha fazla feremon konsantrasyonuna neden olan dallardan birinin seçiminin nedeni, yol seçimindeki rastgele dalgalanmalardır [165]. "İkili köprü deneyi" olarak isimlendirilen bu deneyden yol seçimi belirlemek için basit bir model geliştirilmiştir.



Şekil 5.1. İkili köprü deneyi.

Daha sonra Goss vd. [166], Şekil 5.2.'de görüldüğü gibi köprünün kollarının birini daha uzun yaparak, ikili köprü deneyini geliştirmişlerdir. Köprünün kolları, karıncaların her iki yönde kollardan birine seçebileceği şekilde yerleştirilmiştir.



Şekil 5.2. Karınca deneyinde kullanılan köprü.

Deneyin sonucunda birkaç dakikalık bir geçiş aşamasından sonra karıncaların tümü en kısa köprü kolunu kullanmaya başlamışlardır. Goss vd. [166], kısa yolun seçilmesi olasılığının iki yol arasındaki uzunluk oranı ile arttığını bulmuşlardır. Bu durum Dorigo ve arkadaşları [167] tarafından 'diferansiyel yol uzunluğu etkisi' olarak adlandırılmaktadır. Bu en kısa yol seçme durumunun ortaya çıkması pozitif geri besleme ve farklı yol uzunlukları türünden ifade edilebilir. Ayrıca bu durumun ortaya çıkmasında dolaylı haberleşme formlarının da etkisi bulunmaktadır.

Aslında Arjantin karıncaları yuvalarından yiyeceğe giderken ve tersi yönde hareket ederken geçtikleri yerlere feremon adlı kimyasal bir madde bırakmaktadırlar. Karıncalar köprünün kısa ve uzun kollarının kesiştiği karar verme noktasına geldiklerinde hangi taraftan gideceklerine geçiş yollarına bırakılan feremon maddesini koklayarak olasılıklı bir seçim ile karar vermektedir. Bu durum seçilen bir yolun diğer karıncalar tarafından seçilme olasılığını artıran bir otomatik katalizör görevi görmektedir. Deneyin başlangıcında köprünün kollarının hiçbirinde feremon bulunmamaktadır. Bu nedenle karıncalar hangi taraftan gideceklerinin kararını eşit olasılıkla vermektedirler. Köprü kollarının farklı uzunluklarından dolayı daha kısa yolu seçen karıncalar yiyeceğe en önce varan karıncalar olacaktır. Bu karıncalar yuvaya geri dönerken kesişme noktasına geldiklerinde önceden bıraktıkları feremon izine bağlı olarak daha büyük olasılıkla daha kısa olan yolu seçeceklerdir. Seçilen yola dönüşte yeniden feremon bırakılacak ve bu yol diğer karıncalar için daha fazla tercih edilir hale gelecektir. İşlem ilerlerken kısa olan yolda daha fazla feremon birikecek ve bu yol karıncalar tarafından daha fazla kullanılır hale gelecektir [168].

5.3.2. Yapay Karıncalar

Gerçek karıncalar örnek alınarak modellenen sistemde, yapay karıncalar buldukları rotanın uzunluğunu simgeleyen yapay bir feremon bırakacak ve diğer yapay karıncalar da kısa rotaları bu sayede bularak tercih edeceklerdir. Yapay feremon maddesi miktarının belirli bir hızda buharlaşması da simüle edilerek tercih edilmeyen uzun rotalardaki koku izleri de yavaş yavaş yok olacak ve bu da yapay karıncaların kısa yol dışındaki uzun rotalara sapmasını önleyecektir.

Yapay karıncalarda bazı özellikler gerçek karıncalardan aynen alınmış, gerçek karıncalarda olmayan bazı özellikler ise eklenmiştir. Gerçek karıncalardan alınan özellikler şu şekildedir [169];

- Karıncalar arasında feremon vasıtasıyla kurulan iletişim,
- Feremon miktarının yoğun olduğu yolların tercih edilmesi,
- Kısa yollar üzerinde feremon miktarının daha hızlı artması.

Eklenen özellikler olarak ise şunlar sıralanabilir;

- Gerçek karıncalar gibi tamamen kör değillerdir, ele alınan problem ile ilgili detaylara ulaşabilirler,
- Ayrık zamanlı bir ortamda yaşarlar ve ayrık durumlardan ayrık durumlara hareket ederler,
- Problemin çözümü için oluşturdukları bilgileri hafızada tutabilirler,
- Feremonu oluşturdukları çözüm kalitesinin bir fonksiyonu olarak bırakırlar,
- Feremon bırakmadaki zamanlama ele alınan karınca algoritmasına göre değişebilir.

5.3.3. KKO Algoritmaları ve Sınıflandırılması

Karınca koloni optimizasyonu algoritmaları, karınca sistemi (KS) tabanlı algoritmalar ve Q-öğrenme tabanlı algoritmalar olmak üzere çözüm üretme kurallarına göre ikiye ayrılırlar [170]:

- Karınca Sistemi tabanlı algoritmalar: Bu algoritmalara örnek olarak ilk karınca algoritması olan karınca sisteminin varyasyonları olan karınca yoğunluk, karınca miktar, karınca çevrim algoritmaları verilebilir [10, 161]. Ayrıca Stützle ve Hoos [171] tarafından geliştirilen Max-Min KS de bu grupta yer alır. Feremon bilgisi yerine kenarları araştıran bu sınıftaki tüm algoritmalar, aynı rastgele dönüşüm kuralını kullanmaktadırlar. Ancak kullandıkları güncelleme teknikleri farklıdır.

- Q-Öğrenme tabanlı algoritmalar. Gambardella ve Dorigo [172] tarafından geliştirilen Ant-Q ve yine aynı kişiler tarafından 1997 yılında geliştirilen karınca koloni sistemi (KKS) bu algoritmalara iki örnektir. İki algoritmanın temel farkı kullandıkları güncelleme yöntemlerinde yatmaktadır.

Karınca Sistemi: KS, ilk geliştirilen karınca algoritmasıdır [10]. Karınca sistemi, genel amaçlı bir optimizasyon sistemi olarak düşünülse de, ayrık problemlerin ve esasen gezgin satıcı probleminin çözümüne yönelik geliştirilmiş bir algoritmadır. Gezgin satıcı problemi, n adet şehir verildiğinde, gezgin satıcının her bir şehri bir kez ziyaret etmek şartıyla minimum uzunluklu kapalı bir turu oluşturma problemi olarak tanımlanır.

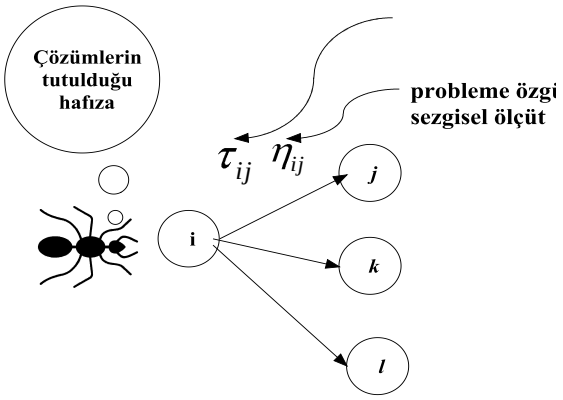
Karınca sistemi içinde geliştirilen ve feremon güncelleme mekanizmaları farklı olan üç karınca sistemi algoritması vardır. Bunlar;

- Karınca yoğunluk,
- Karınca miktar,
- Karıca çevrim.

Karınca sisteminde, gezgin satıcı problemi çözülürken ilk olarak karıncalar rastgele şehirlere yerleştirilirler. Gerçek karıncalardan farklı olarak, yapay karıncaların turları esnasında geçtikleri yolları tuttukları hafızaları vardır. Karıncaların bir şehirden başka bir şehre geçiş olasılıkları yolda biriken feremon miktarına ve öklit uzaklığı kullanılarak hesaplanan seçilebilirliğe bağlı olarak hesaplanır. Karınca k'nın t anında,i şehrinden j şehrine geçiş olasılığı (5.1.) eşitliği ile hesaplanır;

$$P_{ij}^{k}(t) = \begin{cases} \frac{[\tau_{ij}(t)]^{\alpha} * [\eta_{ij}]^{\beta}}{\sum [\tau_{ij}(t)]^{\alpha} * [\eta_{ij}]^{\beta}} & eğer\ j\ izin\ verilen\ değerler\ arasındaysa \\ 0 & aksi\ durumda \end{cases} \qquad (5.1.)$$

Burada, 'j izin verilen değerler' ifadesinden kasıt j şehrinin daha önce uğranmamış şehirlerden biri olup olmadığıdır. τ_{ij}, i ve j şehirleri arasındaki feremon miktarını, η_{ij} ise i şehrinden j şehrine uzaklığın tersi olan seçilebilirliği göstermektedir. α ve β parametreleri ise feremon ve seçilebilirlik arasındaki göreli önemi göstermektedir. Dolayısıyla geçiş olasılığı, seçilebilirlik değeri ve feremon miktarı arasında bir kararı belirler. Yani, seçilebilirlik, yakın şehirlerin daha yüksek ihtimalle seçilmesini destekleyerek bir aç gözlü sezgisel kural tanımlarken, feremon miktarı ise daha fazla trafiğin olduğu hattın tercih edilmesini sağlar. Şekil 5.3. karınca sistemini mantığını vermektedir.



Şekil 5.3. Karınca sistemi mantığı.

Kullanılan KKO algoritmasından bağımsız olarak feremon güncellemede yerel ve küresel olmak üzere iki farklı feremon güncelleme yöntemi vardır. Yerel feremon güncelleme, bir şehir seçildikten sonra, tur oluşturma aşamasında uygulanır ve uygulanması isteğe bağıldır. Küserel feremon güncelleme ise bütün karınca sistemi algoritmalarında mevcuttur, eşitlik (5.2) ile hesaplanır ve algoritmadan algoritmaya farklılık gösterir.

$$\tau_{ij}(t+1) = (1-\rho) * \tau_{ij}(t) + \rho * \sum_{k=1}^{m} \Delta\tau_{ij}^{k}(t) \qquad (5.2)$$

Burada $\tau_{ij}(t+1)$, bir sonraki iterasyonda kullanılacak i-j yolundaki feremon miktarını, $\tau_{ij}(t)$ ise tamamlanan iterasyondan sonra *i-j* yolundaki feremon miktarını göstermektedir. ρ, 0-1 arasında değer alabilen buharlaşma katsayısıdır ve feremon miktarının zamanla buharlaştırılmasında kullanılır. $\sum_{k=1}^{m} \Delta\tau_{ij}^{k}(t)$, k karıncasının i-j hattını kullanması sonucu bu hatta meydana gelen feremon miktarındaki toplam değişimi ifade etmektedir. Bu miktarın hesaplanması karınca miktar, karınca yoğunluk ve karınca çevrim algoritmalarında farklıdır;

- *Karınca Yoğunluk:* Bir karınca *i-j* hattını her kullandığında bir miktar feremon *i-j* hattına bırakılır.

$$\Delta\tau_{ij}^{k}(t) = \begin{cases} 1, & eğer\ k\ karıncası\ i-j\ hattını\ kullanırsa \\ 0, & aksi\ halde \end{cases} \qquad (5.3.)$$

- *Karınca Miktar:* *i-j* hattının arasındaki mesafenin tersiyle orantılı bir miktar feremon bu hatta bırakılır. Dolayısıyla hat ne kadar kısa olursa, depolanan feremon da o derece fazla olacaktır.

$$\Delta\tau_{ij}^{k}(t) = \begin{cases} \frac{1}{d_{ij}}, & eğer\ k\ karıncası\ i-j\ hattını\ kullanırsa \\ 0, & aksi\ halde \end{cases} \qquad (5.4.)$$

d_{ij} : *i-j* hattı arasındaki mesafe

- *Karınca Çevrim:* çözüm kalitesine bağlı olarak *i-j* hattına feremon depolanır. Dolayısıyla oluşan tur ne kadar kısa olursa daha fazla feremon birikecektir.

$$\Delta\tau_{ij}^{k}(t) = \begin{cases} \frac{1}{U_k}, & eğer\ k\ karıncası\ i-j\ hattını\ kullanırsa \\ 0, & aksi\ halde \end{cases} \qquad (5.5.)$$

U_k : k karıncasının oluşturduğu turun uzunluğu

Aşağıda basit karınca sistemi algoritması verilmektedir;

```
begin
     Parametrelere ve feremon miktarlarına başlangıç
    değerlerini ver.
     Repeat
          - Yapay tüm karıncalar için yapay yollar üret
          - Tüm yapay yolların uzunluğunu hesapla
          - Yollara yapıştırılan feremon miktarlarını güncelle
          - Şu ana kadar bulunan en kısa yolu sakla
     Until Maksimum iterasyon sayısı veya önceden tanımlanmış
    bir kriter sağlanana kadar
end
```

Max-Min Karınca Sistemi: Stützle ve Hoos [171] tarafından önerilen Max-Min Karınca Sistemi, KS ile aynı geçiş ve seçim kuralını kullanmasından dolayı KS algoritması ile temelde aynı yapıdadır. Ancak, KS algoritmasına dört yenilik getirmiştir ve böylece karınca sistemi algoritmasından ayrılır [173]. Bu yenilikler şöyle sıralanabilir;

- Max-Min sistemi en iyi turları başarılı bir şekilde bulur. Bunu yalnızca o turdaki veya o ana kadar olan turlardaki en iyi karıncanın feremon bırakmasına izin vererek başarır. Ancak böyle bir strateji erken yakınsamaya neden olabilir.

- Erken yakınsamayı engellemek için feremon miktarları $[\tau_{min}, \tau_{max}]$ aralığında sınırlandırılır.

- Başlangıç feremon miktarları maksimum değerlerine atanır. Böylece araştırmanın başlangıcında turların keşfedilme ihtimali, düşük buharlaşma oranıyla birlikte artar.

- Erken yakınsama durumu veya daha iyi çözümün bulunamaması durumunda feremon miktarları başlangıç değerlerine getirilir.

Karınca-Q Algoritması: Q-öğrenme tabanlı bir karınca koloni optimizasyonu algoritması olan Karınca-Q algoritması 1995 yılında Gambardella ve Dorigo [172] tarafından geliştirilmiştir. Karınca-Q algoritmasında yerel kural güncelleme Q-öğrenmeden esinlenilmiştir. Feremon notasyonu yerini AQ değeri almıştır. Karınca-Q algoritmasının amacı, AQ değerlerini öğrenerek iyi çözümler elde edebilmektir.

Karınca Koloni Sistemi: Karınca koloni sistemi 1997 yılında Dorigo ve Gambardella [169] tarafından karınca sistemi geliştirilerek ortaya atılmıştır. KS'den dört temel farklılığı vardır. Bunlar [165];

- Karınca sistemine göre daha çevik bir seçme kuralı uygular. Böylece karıncaların araştırma kabiliyeti daha başarılı hale gelir. Karınca sistemindeki yol seçme kuralı, açık ve kesin bir keşif davranışının ortaya çıkmasını sağlamak için geliştirilmiştir.

- Feremon buharlaştırma ve feremon bırakma işlemleri sadece o ana kadar bulunan en iyi yolda gerçekleştirilir. Daha kısa bir ifadeyle, programın her tekrarından sonra, sadece o ana kadar bulunan en iyi karıncanın feremon bırakmasına izin verilir.

- Karıncaların kullandıkları bir yolun feremon miktarının bir kısmı her seferinde alternatif yolların aranmasını sağlamak için azaltılır. Bu şekilde farklı bir yerel feremon güncelleme gerçekleştirilir.

- Ziyaret edilecek alternatif şehir listesini kısıtlamak için aday listeler kullanılır. Bu genellikle büyük gezgin satıcı problemlerini çözmekte kullanılan bir veri

yapısıdır. Aday listesi, bir şehirden sonra öncelikle veya en yakın ziyaret edilmesi istenen şehirlerin listesidir. Verilen herhangi bir şehirden ihtimal dahilinde gidilebilecek bütün şehirleri incelemek yerine, öncelikle aday listesindeki ziyaret edilmeyen şehirler incelenir. Aday listesindeki tüm şehirler ziyaret edilmişse diğer şehirler incelenir. Aday listesindeki şehirler, artan uzaklıklarına göre sıraya konulmuşlardır ve liste düzenli olarak taranır.

5.3.4. Sürekli Problemler için Karınca Koloni Algoritmaları

Kesikli problemler için geliştirilen karınca koloni optimizasyonunun sürekli problemler için uygulanması ile ilgili çok az çalışma mevcuttur. En bilinen çalışmalar olarak şunlar sıralanabilir;

- Sürekli Karınca Koloni Optimizasyonu (SKKO)
- Izgara Tabanlı Karınca Koloni Optimizasyonu (ITKKO)
- Pachycondyla Apicalis Optimizasyon Algoritması (API)
- Tur Atan Karınca Koloni Optimizasyonu (TAKKO)

Sürekli Karınca Koloni Optimizasyonu: SKKO algoritması 1996 yılında Wodrich [174] tarafından geliştirilmiştir ve sürekli problemler için geliştirilen ilk karınca algoritmasıdır. Karıncalar yerel ve küresel olmak üzere ikiye ayrılır. Küresel karıncalar küresel arama yaparlar ve toplam karınca sayısının yaklaşık %80'ini oluştururlar. Algoritma başlangıcında bu küresel karıncalar arama uzayına rastgele dağılır. Küresel karıncaların belirledikleri bölgelere uygunluk değeri ile orantılı sayıda yerel karıncalar detaylı arama için gönderilir. Yerel karıncalar küresel karıncalar tarafından uygun bulunan bölgelerde kokuya dayalı arama yaparlar. İyi çözüm bulunduğunda yerel karınca, seçtiği yerin feremon değerini günceller ve daha iyi çözümün bulunduğu noktaya daha çok sayıda yerel karınca gönderilir. Her bölgede arama yapan son yerel karıncanın arama vektörü muhafaza edilir. Eğer son karınca daha iyi bir çözüm bulduysa yeni karınca bu noktadan aramaya başlar. Gelişim sağlanmadıysa yeni karınca rastgele bir yön seçer.

SKKO algoritmasında bir sayaç parametresi yerel karıncaların ne kadar uzağa erişebileceğini belirlemek üzere kullanılır. Çözümünü geliştiren yerel karınca, feremon miktarını artırırken aynı zamanda bu çözümün sayaç değerini de bir azaltır. Bir yönün

sayaç değeri azaldıkça o yönü seçen karıncanın aynı yönden itibaren uzanabileceği mesafe miktarı artar.

Sürekli karınca koloni optimizasyonunda yerel optimumdan kurtulmak için küresel karıncalar geniş alanlara uzanabilme yeteneğine sahiptir. Küresel arama için genetik algoritmadan esinlenen çaprazlama benzeri bir fonksiyon kullanılır. Algoritmada kullanılan küresel arama metodunun karınca sistemi ile ilgisi yoktur [175].

Sürekli problemler için geliştirilen SKKO algoritmasının en büyük dezavantajı karıncaların bir noktayı birden fazla arama hatasına düşmesidir.

Izgara Tabanlı Karınca Koloni Optimizasyonu: Izgara tabanlı KKO algoritması tur atan KKO algoritması ile birlikte Hiroyasu ve arkadaşları [12] tarafından geliştirilmiştir. ITKKO algoritması, araştırma uzayını belirli aralıklarda ızgaralara böler. Yapay karıncalar, bu yapay ızgaralar üzerinde bir düğümden diğerine hareket eder ve sadece düğüm noktasına feremon bırakır. Izgaradaki düğümlere karıncalar gelişigüzel biçimde dağıtılır. Her karınca kendi komşu düğümlerini gezer ve geçtiği her düğüm noktasına feremon bırakır. Karıncaların bir düğümden başka bir düğüme hareketi, olasılık tabanlıdır ve bu olasılık (5.6.) denklemi ile hesaplanır;

$$P_j = \frac{\tau_j^{\alpha}.\eta_j^{\beta}}{\sum_{i=1}^{M} \tau_i^{\alpha}.\eta_i^{\beta}} \tag{5.6.}$$

Izgara tabanlı KKO algoritmasında, karınca geçtiği düğüme her harekette feremon bırakır. Bırakılan feremon miktarı aşağıdaki eşitlik ile hesaplanır;

$$\Delta\tau_j = \frac{Q}{F_j} \tag{5.7.}$$

j noktasına *t* anında bırakılan toplam feremon miktarı ise, *m* adet karınca *j* noktasını kullanıyorsa 5.8. ve 5.9. denklemleri ile hesaplanır.

$$\Delta\tau_j(t) = \sum_{k=1}^{m} \Delta\tau_j(t) \tag{5.8.}$$

$$\tau_j(t) = \tau_j(t-1) + \Delta\tau_j(t) \tag{5.9.}$$

j noktasında feremon buharlaşması ise *n* iterasyon sonra, ρ buharlaşma katsayısına bağlı olarak aşağıdaki şekilde hesaplanır;

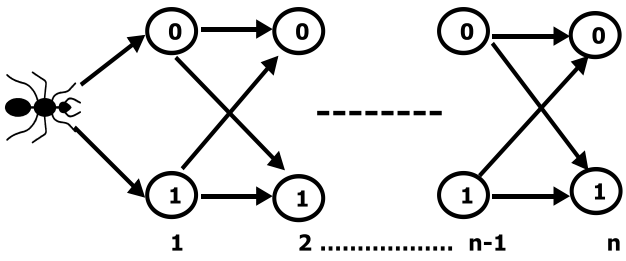
$$\tau_j(t+n) = \rho.\tau_j(t) \qquad (5.10.)$$

Bu işlemler belirlenen bir durdurma kriteri sağlanana kadar devam eder.

Pachycondyla Apicalis Optimizasyon Algoritması: Özel bir karınca türünün davranışına dayanan API algoritması, Monmarche ve arkadaşları [176] tarafından 2000 yılında ortaya atılmıştır. API algoritması karınca tabanlıdır ve bütün karıncalar için küresel bir hafıza kullanılır. En belirgin özelliği aramayı yönlendirmede koku yani feremon kullanılmaması ve çevreye dayalı etkileşim olmamasıdır. Bu nedenle çoğu zaman API algoritması bir karınca modeli olarak görülmemektedir.

API algoritmasında karıncalar önce arama uzayında gelişigüzel dağıtılır ve arama uzayını rastgele yönlerde aramaya başlarlar. Karıncalar yönlendirildikleri noktada yerel aramayı paralel yapıda gerçekleştirirler ve her karınca bulduğu en iyi noktayı hafızasında tutar. Daha sonra bütün karıncalar bilgi değişimi yapar ve yuva olarak isimlendirilen koloni merkezi, bulunan en iyi noktaya taşınır ve bu yeni nokta etrafında arama süreci tekrar başlar.

Tur Atan Karınca Koloni Optimizasyonu: Tur atan karınca koloni optimizasyonu algoritması, optimizasyon problemlerinde ikili diziler şeklinde kodlanan sürekli değişkenleri ele almak için Hiroyasu vd. [12] tarafından tasarlanmıştır. Temel karınca koloni optimizasyonu algoritmasından farklı olarak TAKKO algoritmasında her bir çözüm ikili bitlerden oluşan bir dizi ile gösterilir. Her karınca, çözüm parametrelerinin birbirine seri olarak eklendiği ikili bir dizisi üzerinde hareket ederek 1 ve 0'lardan oluşan bir çözüm dizisi üretir. Tur atan karınca koloni optimizasyonu algoritmasının temel prensibi Şekil 5.4.'de gösterilmektedir.



Şekil 5.4. Tur atan karınca koloni optimizasyonu temel prensibi.

Karıncalar bir bitin değerinin 0 veya 1 olmasına karar vermeye çalıştıklarında sadece feremon bilgisini kullanırlar. Bir karınca çözümünü tamamladıktan ve çözümün kalite değeri hesaplandıktan sonra, çözümü oluşturan alt yollara yapıştırılacak olan yapay feremon hesaplanır. Kolonideki tüm karıncalar yollarını tamamladıktan ve yollara yapıştırılacak olan feremon miktarı hesaplandıktan sonra, alt-yolların bitleri arasındaki feremon miktarları güncellenir [177].

Feremon bitler arasındaki yollara değil bitleri temsil eden ayrık noktalara bırakılır. TAKKO algoritmasında seçilebilirlik parametresi kullanılamaz. Bu nedenle karınca hareketi sırasında her adımda 0 veya 1 noktasını seçmeden önce, sadece bu noktalara bırakılan feremon miktarını kullanarak karar verir.

TAKKO algoritmasında, 0 ve 1 $(0 \rightarrow 1)$ bitleri arasındaki alt yolun tercih edilmesi olasılığının hesaplanmasında (5.11.) eşitliği kullanılır.

$$p_{01} = \frac{\tau_{01}}{\tau_{01}+\tau_{00}} \tag{5.11.}$$

Burada p_{01}, $(0 \rightarrow 1)$ alt yolunun seçilme olasılığını temsil eder τ_{00} ve τ_{01} sırasıyla $(0 \rightarrow 0)$, $(0 \rightarrow 1)$ alt yollarına yapışık yapay feremon maddesi miktarıdır. Bu olasılık fonksiyonu ile bir seçme mekanizması kullanılarak tur tamamlanır. Tur tamamlandıktan sonra oluşturulan çözüm problem için değerlendirilir ve probleme özel belirlenen bir kalite fonksiyonuna göre o çözümün kalite değeri (F_k) hesaplanır. Pozitif bir sabit olan Q ve hesaplanan kalite değerine bağlı olarak çözümü üreten karıncanın geçtiği noktalara feremon bırakılır. Tur tamamlanırken seçilmiş olan j noktasına bırakılacak yapay feremon maddesi (5.12.) eşitliği ile hesaplanır.

$$\Delta\tau_{01}^{k}(t,t+1) = \begin{cases} \frac{Q}{F_k} & eğer\ k\ karıncası\ alt\ yol\ (0-1)'den\ geçerse \\ 0 & aksi\ halde \end{cases} \tag{5.12.}$$

Burada $\Delta\tau_{01}^{k}$ k. yapay karınca tarafından $(0 \rightarrow 1)$ alt yoluna yapıştırılan feremon maddesi miktarıdır; M tane karınca araştırma işlemini tamamladıktan ve yolları belirledikten sonra $(0 \rightarrow 1)$ alt yoluna, t ve $(t+1)$ zaman aralığında atanacak feremon maddesi miktarı aşağıdaki bağıntı ile hesaplanır:

$$\Delta\tau_{01}(t,t+1) = \sum_{k=1}^{N} \Delta\tau_{01}^{k}(t,t+1) \tag{5.13.}$$

Aynı alt yolun (*t+1*) anındaki feremon miktarı (5.14.) eşitliği ile belirlenir:

$$\tau_{01}(t+1) = \rho\tau_{01}(t) + \Delta\tau_{01}(t, t+1) \quad (5.14)$$

Burada ρ, buharlaşma parametresi olarak adlandırılır ve $(1-\rho)$, feremon maddesinin buharlaşma miktarını gösterir $(0 \leq \rho < 1)$.

Çözüm üretirken, yukarıda anlatılan prensipleri izleyen TAKKO algoritmasının temel adımları aşağıdaki şekildedir:

---

begin

Parametrelere ve feremon miktarlarına başlangıç değerlerini ver.

Repeat

- *Yapay tüm karıncalar için yapay yollar üret*
- *Tüm yapay yolların kalitesini hesapla*
- *Yollara yapıştırılan feremon miktarlarını güncelle*
- *Şu ana kadar bulunan en iyi yolu sakla*

Until *Maksimum iterasyon sayısı veya önceden tanımlanmış bir kriter sağlanana kadar*

end

5.4. Sonuç

Bu bölümde, geliştirilen algoritmanın kural çıkarım bölümünde kullanılan karınca koloni optimizasyonu algoritması ele alınmıştır. Sürü zekası, KKO algoritması, gerçek karınca kolonileri ve yapay karıncalar hakkında bilgi verildikten sonra KKO algoritmaları sınıflandırılarak açıklanmıştır. Son olarak ise KKO algoritmaların özelliklerine değinilmiştir.

Bu bölümde verilen bilgiler geliştirilen algoritmanın kural çıkarım bölümü için bir alt yapı oluşturmaktadır.

6. BÖLÜM
YAPAY SİNİR AĞLARINDAN KARINCA KOLONİ OPTİMİZASYONU İLE SINIFLANDIRMA KURALLARI ÇIKARIMI

6.1. Giriş

Bu bölümde, tez çalışmasında geliştirilen TAKKOYSA-sınıflandırıcısı anlatılmaktadır. İlk olarak geliştirilen algoritmanın genel yapısı ve işleyişi açıklandıktan sonra, temel adımları verilmiştir. Algoritmada kullanılan veri gösterimi örnekle açıklanmıştır. Son olarak ise TAKKOYSA-sınıflandırıcısının YSA'ların eğitimi, TAKKO ile kural üretimi, kural indirgeme, kural basitleştirme ve dilsel kurallara dönüştürme olarak dörde ayrılan ana bölümleri alt bölümlerine ayrılarak detaylandırılmıştır.

6.2. Genel Yapısı ve İşleyişi

Bir yapay sinir ağı karmaşık girdi/çıktı ilişkilerini belirleme ve gösterme kabiliyetine sahip güçlü bir veri modelleme aracıdır. YSA'lar insan beynine, bilgiyi öğrenme sayesinde elde etmesi ve bilginin sinaptik ağırlıklar olarak bilinen içsel-nöron bağlantılarında depolanması şeklinde iki yönden benzerler. Sınıflandırma, tanıma ve genellemede çok iyi araçlar olmalarına rağmen, elde ettikleri bilgileri kullanıcıya sunamaz ve dilsel kurallar şeklinde ifade edemezler. Bu nedenle son yıllarda birçok araştırmacı, yapay sinir ağlarını dilsel kurallar şeklinde ifade eden modeller geliştirme eğilimindedirler. Bu amacı gerçekleştirmenin yolu ise eğitilmiş yapay sinir ağlarından kurallar çıkarmaktır.

Eğitilmiş yapay sinir ağlarından kural çıkarımı Craven ve Shavlik [134] tarafından "eğitilmiş bir yapay sinir ağı ve bu ağı eğitmekte kullanılan örnekler verildiğinde, ağların özlü ve doğru sembolik tanımlarının üretilmesi" olarak tanımlanır. YSA'lardan kural çıkarımı son kullanıcıların sinir ağının gerçekte ne öğrendiğini ve nasıl çalıştığını

anlamasına yardımcı olur. Ayrıca yapay sinir ağlarının iç parametrelerinden kurallar çıkarımı YSA'ların bazı dezavantajlarının üstesinden gelir.

Bir yapay sinir ağıyla elde edilen bilgi, YSA'nın hem mimarisiyle hem de aktivasyon fonksiyonlarıyla ilgili olan bağlantı ağırlıklarında kodlanır [8]. Bu bağlamda, yapay sinir ağlarından bilgi çıkarımı süreci genel olarak ya bağlantı ağırlıkları değerlerine ya da gizli birim aktivasyon değerlerine dayanan algoritmaların kullanımını gerektirir. Bunun için tasarlanan algoritmalara genel olarak "*YSA'dan kural çıkarımı için algoritmalar"* denilir [7].

Geliştirilen kural çıkarım algoritması önceden eğitilmiş YSA'lardan kural çıkarımı problemine odaklanmıştır ve aktivasyon fonksiyonu üzerine herhangi bir tahmin yapmamaktadır. Belirli sınıflara ait kurallar çıkarmak için sadece ağırlık değerleri kullanıldığından analizci bir yaklaşımdır.

Önerilen TAKKOYSA-sınıflandırıcısı ikili veri kümeleri üzerinde çalışmaktadır. İkili gösterim ile eğitilmiş YSA'lardan elde edilen ağırlıklar kullanılarak hangi bağlantı ağırlıklarının çıktı düğümünü en iyilediğine yönelik sınıflandırma kuralları çıkarılabilmektedir. Geliştirilen kural çıkarım algoritması ile sürekli ve kesikli değişken içeren veri kümelerinden sınıflandırma kuralları çıkarabilmek için veri kümelerinin öncelikle ikili forma dönüştürülmesi gerekmektedir.

Verinin gösterimi ikili olduğu için, bu yapıya en uygun KKO algoritması Bölüm 5'de de anlatıldığı gibi tur atan karınca koloni optimizasyonu algoritmasıdır. Algoritmada bazı adaptif değişiklikler yapılarak, TAKKO algoritması geliştirilen yaklaşımın sınıflandırma kural üretimi bölümünde kullanılmıştır.

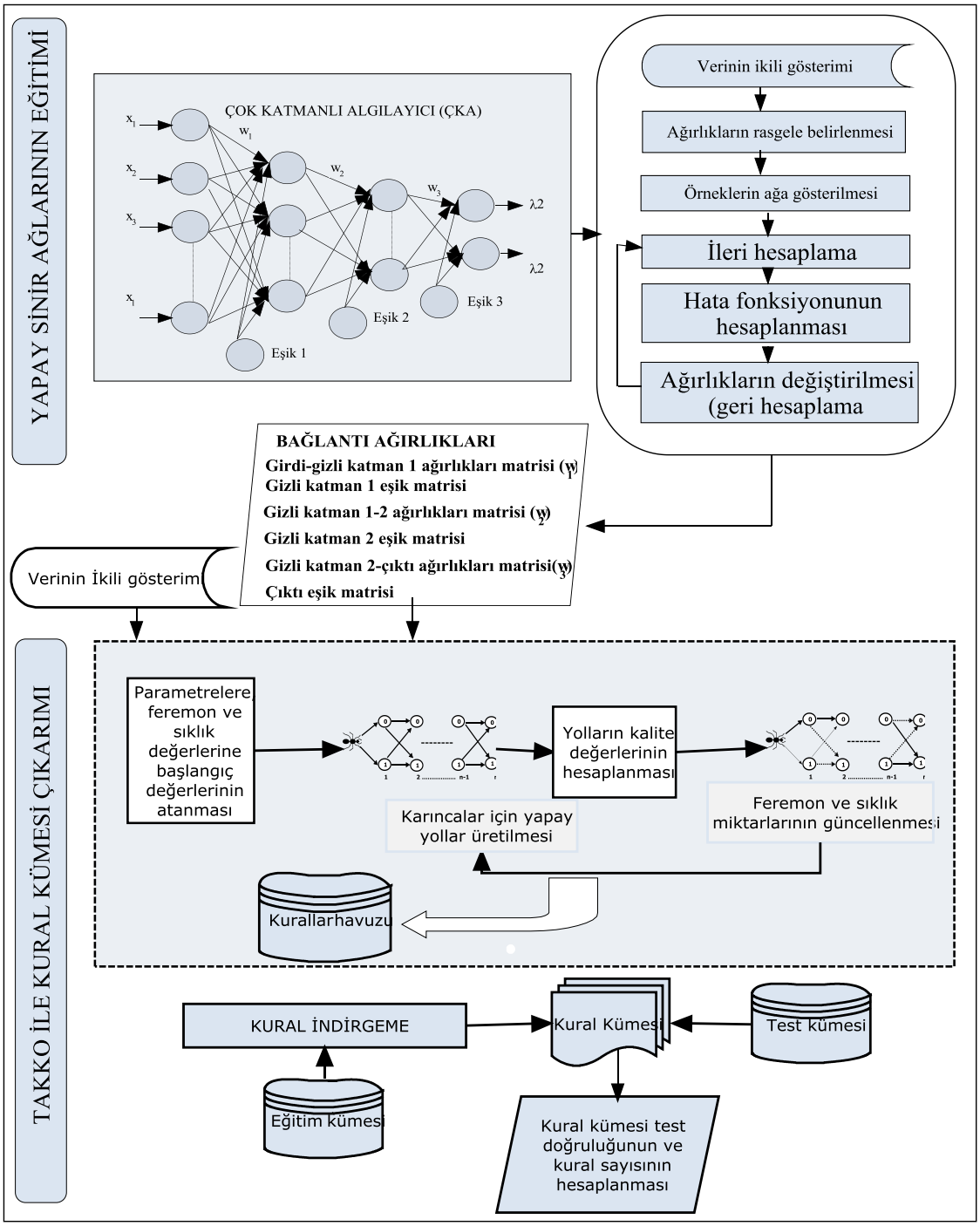
Yapılan çalışmada sinir ağı olarak, en yaygın kullanıma sahip yapay sinir ağlarından biri olan çok katmanlı algılayıcı modeli ele alınmıştır. Daha önce de değinildiği gibi ÇKA özellikle sınıflandırma, tanıma ve genelleme gibi özel problemler için kullanışlıdır. Tez çalışmasının da amacı eğitilmiş yapay sinir ağlarından sınıflandırma kuralları çıkarmak olduğu için ÇKA bu amaca en uygun modeldir.

Eğitilmiş sinir ağlarından kural çıkarımı için geliştirilen algoritma dört bölümden meydana gelmektedir. Bunlar;

1. Çok katmanlı algılayıcıların eğitimi ve ağırlıkların elde edilmesi,
2. Elde edilen ağırlıklara ve ikili veri kümelerine bağlı olarak TAKKO algoritması ile sınıflandırma kurallarının üretilmesi,
3. Kural indirgeme prosedürü,
4. Kural basitleştirme ve dilsel kurallara dönüştürme.

TAKKOYSA-sınıflandırıcısının ilk bölümünde ÇKA modeli eğitilmekte ve buradan elde edilen ağırlıklar TAKKO ile kural çıkarım bölümüne girdi teşkil etmektedir. Bu bölümde belirli bir kaliteyi sağlayan kurallar kural havuzunda, kural indirgeme işlemi için saklanmaktadır. Son olarak kural indirgeme bölümünde, veri kümesini en iyi yansıtan kural listesi oluşturulmaktadır. Önerilen algoritma ile çıkarılan kurallar "EĞER…..O HALDE" formunda sınıflandırma kurallarıdır. Kuralın önde gelen kısmı (EĞER) girdi parametreleriyle ilgili önermeleri, sonda gelen kısmı (O HALDE) ise tahminlenen sınıfı içerir.

Geliştirilen kural çıkarım algoritması Intel® Core ™ 2 Duo CPU 2,4 GHz 2 GB RAM bilgisayar kullanılarak "DELPHİ 7" programlama dilinde kodlanmıştır. TAKKOYSA-sınıflandırıcısının temel yapısı Şekil 6.1.'de gösterilmektedir.



Şekil 6.1. TAKKOYSA-sınıflandırıcısının temel yapısı.

6.3. TAKKOYSA-sınıflandırıcısının Temel Adımları

Çok katmanlı algılayıcıların eğitilmesi, TAKKO ile kural üretimi, kural indirgeme ve kural basitleştirme-dilsel kurallara dönüştürme prosedürlerinin birleştirilmesi ile geliştirilen TAKKOYSA-sınıflandırıcısının temel adımları şu şekilde özetlenebilir;

Begin
- *Parametre değerlerinin atanması*
- *Ağırlıkların başlangıç değerlerinin rastgele üretilmesi*

Repeat
- *İleri hesaplama*
- *Hata hesaplama*
- *Geri hesaplama*

Until *maksimum iterasyon sayısı*
- *Ağırlık matrislerinin elde edilmesi*
- *Feremon ve sıklık miktarlarının başlangıç değerlerinin atanması*

Repeat

Begin

Repeat
- *Tüm yapay karıncalar için feremon ve sıklık miktarlarına göre yapay yolların üretilmesi*
- *(6.27.) eşitliğinde verilen olasılık fonksiyonuna göre bir yol seçilmesi*
- *Karıncaların kalite ve uygunluk değerlerinin hesaplanması*
- *YSA çıktı fonksiyon değerine göre yollara yapışık feremon ve sıklık miktarlarının güncellenmesi*
- *Kalite değeri eşik değerinden büyük olan çözümlerin kural indirgeme prosedürü için kural havuzunda saklanması*

Until *maksimum iterasyon sayısı*

End

Until *sınıf sayısı*
- *Kural listesinin boş olarak belirlenmesi*

Repeat
- *Kural havuzunda en yüksek uygunluk değerine sahip olan kuralın belirlenmesi*
- *Kuralın sağladığı örneklerin eğitim kümesinden çıkarılması*
- *Çözümün kural olarak saklanması ve kural listesine eklenmesi*

Until *tüm eğitim örneklerinin sağlanması veya kural havuzunun boşalması*
- *Kural listesinin test veri kümesine uygulanması ve performans ölçütlerinin hesaplanması*

Repeat
- *Çözüm kümesindeki ilk kuraldan başlanması*
- *Gereksiz değişkenlerin çıkarılması (kuralda alt-dizi bit değeri aynı olan değişkenler)*
- *"VE" ve "VEYA" mantıksal operatörlerinin sırasıyla aynı değişkenin mevcut değerlerini ve farklı değişkenleri birleştirmek için kullanılması.*

Until *kural kümesinin sonuna kadar.*

End.

6.4. Veri Gösterimi

TAKKOYSA-sınıflandırıcısında veri kümelerindeki her bir veri, ikili diziler şeklinde kodlanmış ve çok katmanlı algılayıcılara girdi olarak gösterilmiştir. Verinin bu şekilde ikili gösterimi ilk olarak Elalfi ve arkadaşları [149] tarafından eğitilmiş yapay sinir ağlarından genetik algoritmalar ile kural çıkarımı için kullanılmıştır.

İkili yapı şu şekilde çalışmaktadır: veri kümesinin N değişkenden oluştuğu varsayılsın. Bu durumda her bir değişken A_n (n=1, 2, 3, …., N); m_n, A_n değişkeninin olası değerlerini göstermek üzere sabit uzunlukta bir ikili alt-diziye (a_1, a_2 , a_3,….., a_{m_n}) kodlanabilir. İkili kodlama gösterimine göre, m_n elemanın sadece aktif hale gelen bir elemanı "1" değerini alabilir ve diğer bütün elemanlar "0" değerini alacaktır. Bu nedenle, girdi vektörünün uzunluğu *GU* şu şekilde belirlenir;

$$GU = \sum_{n=1}^{N} m_n \tag{6.1.}$$

k sınıf sayısını göstermek üzere, çıktı sınıf vektörü CU_k (*k=1, 2, 3, …, K*) da aynı şekilde sabit uzunlukta ikili diziye kodlanabilir. Girdi değişkeni alt-dizilerinde olduğu gibi, çıktı sınıf vektöründeki k elemandan yine sadece biri, eğer çıktı vektörü k sınıfına dahilse "1" değerini alabilecek ve vektördeki diğer tüm elemanlar sıfır olacaktır. Bu nedenle çok katmanlı algılayıcının çıktı katmanındaki çıktı düğüm sayısı K olacaktır. Çıktı sınıfı vektörü ise (6.2.) eşitliği ile gösterilecektir;

$$CU_k = \{S_1, S_2, S_3, \dots\dots, S_k\} \tag{6.2.}$$

Geliştirilen algoritmada, bu ikili gösterim şekli sadece çok katmanlı algılayıcıların girdilerinin gösteriminde değil; aynı zamanda algoritmanın kural üretim bölümünde kural gösteriminde de kullanılmıştır. *GU* ikili girdi vektörleri sayısını ve K ikili çıktı sınıf vektör sayısını gösterdiğine göre kural uzunluğu (*KU*) bu durumda (6.3.) eşitliği ile hesaplanacaktır.

$$KU = GU + K \tag{6.3.}$$

Aşağıda TAKKOYSA-sınıflandırıcısında ikili gösterimin nasıl uygulandığı ile ilgili bir örnek verilmektedir. Ele alınan veri kümesi az sayıda veri ve örnek içermesinden; ayrıca basit olmasından dolayı literatürde yer alan "Tenis Oynama" veri kümesidir. Veri kümesi, {hava durumu, sıcaklık, nem, rüzgar} olmak üzere dört kategorik girdi

değişkeni ve tenis oynama kararının yer aldığı, {oyna, oynama} şeklinde iki değer alabilen bir çıktı değişkeni içermektedir. Orijinal veri kümesi Tablo 6.1.'de gösterilmektedir.

Tablo 6.1. Tenis oynama veri kümesi.

Gün	Hava durumu	Sıcaklık	Nem	Rüzgar	Karar
G1	Güneşli	Sıcak	Yüksek	Hafif	Oynama
G2	Güneşli	Sıcak	Yüksek	Sert	Oynama
G3	Bulutlu	Sıcak	Yüksek	Hafif	Oyna
G4	Yağmurlu	Ilık	Yüksek	Hafif	Oyna
G5	Yağmurlu	Soğuk	Normal	Hafif	Oyna
G6	Yağmurlu	Soğuk	Normal	Sert	Oynama
G7	Bulutlu	Soğuk	Normal	Sert	Oyna
G8	Güneşli	Ilık	Yüksek	Hafif	Oynama
G9	Güneşli	Soğuk	Normal	Hafif	Oyna
G10	Yağmurlu	Ilık	Normal	Hafif	Oyna
G11	Güneşli	Ilık	Normal	Sert	Oyna
G12	Bulutlu	Ilık	Yüksek	Sert	Oyna
G13	Bulutlu	Sıcak	Normal	Hafif	Oyna
G14	Yağmurlu	Ilık	Yüksek	Sert	Oynama

Tablo 6.1.'den de görülebileceği gibi hava durumu değişkeni {güneşli, bulutlu, yağmurlu} olmak üzere üç; sıcaklık değişkeni {sıcak, ılık, soğuk} olmak üzere üç; nem değişkeni {yüksek, normal} olmak üzere iki ve rüzgar değişkeni {hafif, sert} olmak üzere yine iki farklı değer içermektedir. Dolayısıyla geliştirilen algoritma için toplam 3+3+2+2=10 girdi vektörü (GU=10) olacaktır. Çıktı değişkeni ise {oyna, oynama} olmak üzere iki sınıf (CU_k=2) içermektedir. Sonuç olarak çıktı sınıf vektörü sayısı ise 2 olacaktır. Tablo 6.2. tenis oynama veri kümesi için elde edilen ikili girdi ve çıktı sınıf vektörlerini göstermektedir.

Tablo 6.2. Tenis oynama veri kümesi ikili değişkenler.

Hava Durumu			Sıcaklık			Nem		Rüzgar		Karar	
Güneşli	Bulutlu	Yağmurlu	Sıcak	Ilık	Soğuk	Yüksek	Normal	Hafif	Sert	Oyna	Oynama
a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	C_1	C_2

Veri kümesi ikili formda kodlanırken, bu ikili girdi ve çıktı sınıf vertörleri alt-dizilerinden yalnızca bir tanesi "1" değerini alabilmekte, diğerleri "0" olmaktadır. Tablo 6.3. tenis oynama değişkenlerinin ikili formda gösterim sonucu ifade ettikleri değerleri göstermektedir.

Tablo 6.3. Tenis oynama veri kümesi değişkenlerinin ikili ifadesi.

Hava Durumu

a_1	a_2	a_3	Durum
1	0	0	Güneşli
0	1	0	Bulutlu
0	0	1	Yağmurlu

Sıcaklık

a_4	a_5	a_6	Durum
1	0	0	Sıcak
0	1	0	Ilık
0	0	1	Soğuk

Nem

a_7	a_8	Durum
1	0	Yüksek
0	1	Normal

Rüzgar

a_9	a_{10}	Durum
1	0	Hafif
0	1	Sert

Sonuç

C_1	C_2	Karar
1	0	Oynama
0	1	Oyna

Girdi ve çıktı değişkenlerinin yukarıda anlatıldığı şekilde hepsi ikili forma dönüştürülerek çok katmanlı algılayıcılara girdi olarak sunulacak hale getirilmektedir. Tablo 6.4. Tenis Oynama veri kümesinin ikili kodlanmış halini göstermektedir.

Tablo 6.4. Tenis oynama veri kümesi ikili gösterim.

	Hava durumu, m_1=3			Sıcaklık m_2=3			Nem m_3=2		Rüzgar,m_4=			Tenis oynama	
G_m	Güneşli	Bulutlu	Yağmurlu	Sıcak	Ilık	Soğuk	Yüksek	Normal	Hafif	Sert	C_m	Oynama	Oyna
	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}		C_1	C_2
G_1	1	0	0	1	0	0	1	0	1	0	C_1	1	0
G_2	1	0	0	1	0	0	1	0	0	1	C_2	1	0
G_3	0	1	0	1	0	0	1	0	1	0	C_3	0	1
G_4	0	0	1	0	1	0	1	0	1	0	C_4	0	1
G_5	0	0	1	0	0	1	0	1	1	0	C_5	0	1
G_6	0	0	1	0	0	1	0	1	0	1	C_6	1	0
G_7	0	1	0	0	0	1	0	1	0	1	C_7	0	1
G_8	1	0	0	0	1	0	1	0	1	0	C_8	1	0
G_9	1	0	0	0	0	1	0	1	1	0	C_9	0	1
G_{10}	0	0	1	0	1	0	0	1	1	0	C_{10}	0	1
G_{11}	1	0	0	0	1	0	0	1	0	1	C_{11}	0	1
G_{12}	0	1	0	0	1	0	1	0	0	1	C_{12}	0	1
G_{13}	0	1	0	1	0	0	0	1	1	0	C_{13}	0	1
G_{14}	0	0	1	0	1	0	1	0	0	1	C_{14}	1	0

Aynı ikili gösterim şekli daha önce de değinildiği gibi kural gösteriminde de kullanılmıştır. Veri kümesinin kodlanmasından farklı olarak kural gösteriminde, her bir değişken alt-dizilerinden yalnız biri "1" değerini almamakta; birden fazla alt-dizi elemanı "1" değerini alabilmekte veya hepsi "0" veya "1" olabilmektedir. Tablo 6.5. örnek bir kuralı göstermektedir.

Tablo 6.5. Örnek bir kural gösterimi.

Hava Durumu, m_1=3			Sıcaklık, m_2=3			Nem,m_3=2		Rüzgar,m_4=2		Tenis Oyna	
Güneşli	Bulutlu	Yağmurlu	Sıcak	Ilık	Soğuk	Yüksek	Normal	Hafif	Sert	Hayır	Evet
a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	C_1	C_2
1	1	0	1	0	1	1	0	0	1	1	0

Dilsel kural: EĞER hava durumu güneşli **VEYA** bulutlu **VE** sıcaklık sıcak **VEYA** soğuk **VE** nem yüksek **VE** rüzgar sert ise **O HALDE** tenis oynama.

Bu ikili veri gösterim şekli ikili veya kesikli değişken içeren veri kümelerine doğrudan uygulanabilir. Ancak sürekli değişken içeren veri kümelerinde, verilerin ikili forma dönüştürülmeden önce kesiklendirilmesi gerekmektedir.

6.5. Geliştirilen Kural Çıkarım Algoritması

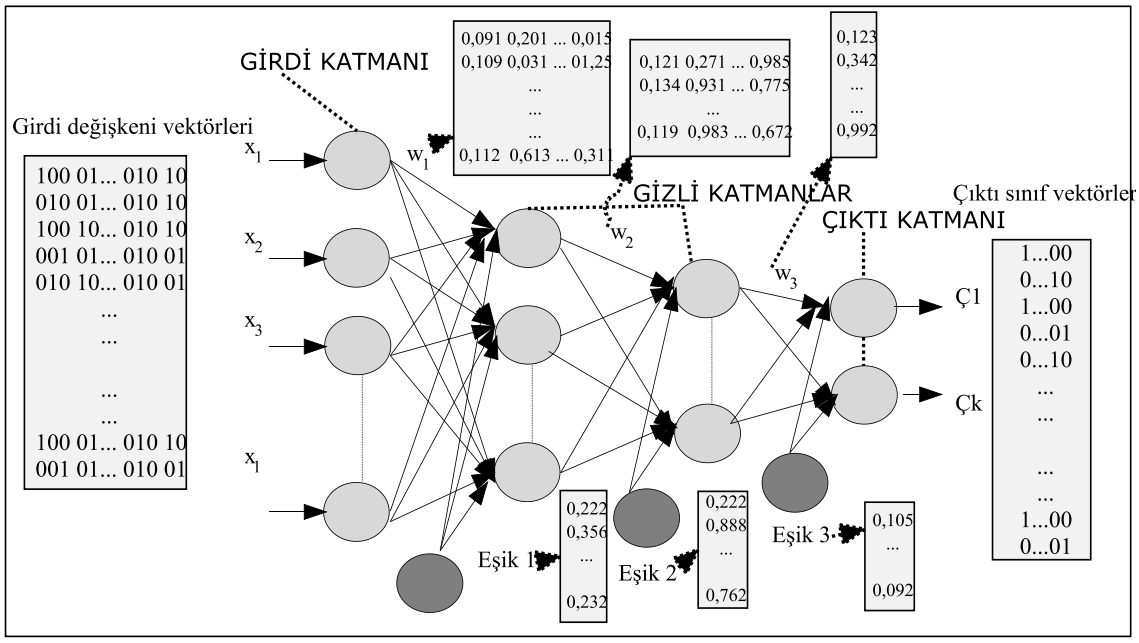
Karınca koloni optimizasyonu ile yapay sinir ağlarından sınıflandırma kuralları çıkarabilmek için öncelikle yapay sinir ağlarının yani çok katmanlı algılayıcıların ikili girdi ve çıktılarla eğitilerek hatayı minimum yapan ağırlık kümelerinin elde edilmesi gerekmektedir. İlerleyen bölümde geliştirilen algoritmanın çok katmanlı algılayıcıların eğitilmesi aşamasında yer alan adımlar detaylı olarak anlatılmaktadır.

6.5.1. Yapay Sinir Ağlarının Eğitimi

Geliştirilen karınca koloni optimizasyonu ile yapay sinir ağlarından kural çıkarımı algoritmasında, yapay sinir ağı modeli olarak Rumelhart ve arkadaşları [130] tarafından geliştirilen çok katmanlı algılayıcı modeli kullanılmıştır. ÇKA modeli, özellikle sınıflandırma, tanıma ve genelleme yapmayı gerektiren problemler için bir çözüm aracıdır [115].

Bu model en küçük kareler yöntemine dayanan delta öğrenme kuralını kullanmaktadır. Temel amacı ağın beklenen çıktısı ile ürettiği çıktı arasındaki hatayı en aza indirmektir. Delta kuralı "*ileri doğru hesaplama*" ve "*geriye doğru hesaplama*" olmak üzere iki safhadan oluşur. İlk safhada ağın çıktısı hesaplanırken ikinci safhada hataya göre ağırlıklar değiştirilir.

Geliştirilen algoritmada daha önce de değinildiği gibi veri gösteriminde ikili kodlama kullanılmıştır. İkili forma dönüştürülen veri kümeleri örnekleri, çok katmanlı algılayıcılara girdi olarak gösterilerek, hatayı en küçükleyen ağırlıklar kural çıkarım prosedürü için saklanmıştır. Şekil 6.2. geliştirilen algoritmada kullanılan üç katmanlı bir çok katmanlı algılayıcı modeline gösterilen örnek ve ağırlık kümeleri yapılarını göstermektedir.



Şekil 6.2. Kullanılan ÇKA örnek yapısı.

TAKKOYSA-sınıflandırıcısında çok katmanlı algılayıcıların eğitilerek, ağırlıkların elde edilmesine yönelik adımlar şu şekildedir;

ADIM 1: ***Ağın topolojik yapısının ve öğrenme parametrelerinin belirlenmesi:***

Gizli katman sayısı, gizli katman(lar)daki işlem elemanı sayısı, öğrenme katsayısı, momentum katsayısı ve iterasyon sayısı parametreleri tur atan karınca koloni optimizasyonu algoritması parametreleri ile birlikte deney tasarımı yapılarak belirlenmiştir. Deney tasarımı ve parametre düzeyleri 7. Bölümde yer alan deney tasarımı ve analizler başlığı altında verilmektedir.

Daha önce de değinildiği gibi çok katmanlı algılayıcılarda en çok tercih edilen aktivasyon fonksiyonu, türevinin alınabilir olmasından dolayı sigmoid aktivasyon fonksiyonudur. TAKKOYSA-sınıflandırıcısında da aktivasyon fonksiyonu olarak sigmoid aktivasyon fonksiyonu kullanılmıştır.

ADIM 2: ***Ağırlıkların başlangıç değerlerinin atanması:***

İşlem elemanlarını yani nöronları birbirine bağlayan ağırlık değerlerinin ve eşik değer ünitelerinin ağırlıklarının başlangıç değerleri 0-1 aralığında rastgele üretilmektedir.

***ADIM 3:* İleri hesaplama:**

Önerilen algoritmada, eğitim kümesindeki ikili örnekler girdi katmanında ağa gösterilir. Girdi katmanında herhangi bir bilgi işleme olmaz yani girdiler hiçbir değişiklik olmadan gizli katmana gönderilir. Girdi katmanındaki l. işlem elemanının çıktısı C_l^i, I_l *l.* işlem elemanına girdiyi göstermek üzere (6.4.) eşitliği ile hesaplanır.

$$C_l^i = I_l \tag{6.4.}$$

Gizli katmanlardaki işlem elemanları girdi katmanındaki tüm işlem elemanlarından gelen bilgileri bağlantı ağırlıkları (w_{lj} *l*=1….girdi işlem elemanı sayısı, *j*=1…..gizli katman işlem eleman sayısı) etkisi ile alır. Gizli katmandaki işlem elemanlarına gelen net girdi toplama fonksiyonu ile (6.5.) eşitliğindeki gibi hesaplanır.

$$NET_j^a = \sum_{l=1}^{n} w_{ij} C_j^i \tag{6.5.}$$

j. gizli katmanın çıktısı ise (6.5.) eşitliği ile hesaplanan net girdinin sigmoid aktivasyon fonksiyonundan geçirilmesi ile hesaplanır. Sigmoid aktivasyon fonksiyon kullanılarak çıktı (6.6.) eşitliği ile hesaplanır. Burada β_j^a, gizli katmanın *j.* elemanına bağlanan eşik değer elemanını göstermektedir.

$$C_j^a = \frac{1}{1+e^{-(NET+\beta)}} \tag{6.6.}$$

Gizli ve çıktı katmanının işlem elemanlarının çıktıları aynı şekilde kendilerine gelen NET girdinin hesaplanması ve sigmoid aktivasyon fonksiyonundan geçirilmesi vasıtasıyla belirlenir. Çıktı katmanından çıkan değerler bulununca, ağın ileri doğru hesaplama işlemi tamamlanmış olur.

***ADIM 4:* Gerçekleşen ile beklenen çıktının karşılaştırılması:**

Ağın ürettiği hata değeri gerçekleşen ile beklenen çıktı kullanılarak hesaplanır. Amaç bu hatanın düşürülmesidir. Bu nedenle, geriye doğru hesaplama yapılırken bu hata ağın ağırlık değerlerine dağıtılarak bir sonraki iterasyonda hatanın azaltılması sağlanır. Çıktı katmanındaki k. işlem elemanı için oluşan hata (6.7.) eşitliği ile hesaplanır. Burada d_k beklenen çıktıyı, c_k ise gerçekleşen çıktıyı ifade etmektedir. Bu bir işlem elemanı için

oluşan hatadır. Çıktı katmanında oluşan toplam hatayı bulmak için (6.8.) eşitliği kullanılır.

$$E_k = (d_k - c_k) \tag{6.7.}$$

$$E(t) = \frac{1}{2}\sum_{k=1}^{K} E_k^2 \tag{6.8.}$$

<u>ADIM 5:</u> Geri hesaplama:

Üretilen hatanın en küçüklenmesi için ağırlıklar geriye doğru hesaplanarak değiştirilir. Bunun için hata, kendisine neden olan işlem elemanlarına dağıtılır. Ağın ağırlıklarını değiştirmek için iki durum söz konusudur: gizli katman-çıktı katmanı arasındaki ağırlıkların değiştirilmesi, gizli katmanlar arası veya gizli katman-girdi katmanı arasındaki ağırlıkların değiştirilmesi.

λ öğrenme katsayısını, α momentum katsayısını, δ_k k. çıktı ünitesinin hatasını göstermek üzere, gizli katmandaki j. işlem elemanını çıktı katmanındaki k. işlem elemanına bağlayan bağlantının ağırlığındaki değişim miktarı ΔD^a (6.9.) eşitliği ile hesaplanır.

$$\Delta D_{jk}^a(t) = \lambda \delta_k C_j^a + \alpha \Delta D_{jk}^a(t-1) \tag{6.9.}$$

k. çıktı ünitesinin hatası ise (6.10.) eşitliğindeki formül ile hesaplanır;

$$\delta_k = f'(NET) E_k \tag{6.10}$$

Bu eşitlikte yer alan f'(NET) aktivasyon fonksiyonun türevini ifade etmektedir. Geliştirilen algoritmada aktivasyon fonksiyonu olarak sigmoid fonksiyon kullanıldığından dolayı, bu fonksiyonun türevi (6.11.) eşitliği ile hesaplanır;

$$\delta_k = C_k(1 - C_k).E_k \tag{6.11.}$$

Ağırlıkların t. iterasyondaki yeni değerleri (6.12.) eşitliği ile hesaplanır. β_k^c, k. çıktı ünitesine bağlanan eşik değer elemanını göstermek üzere, eşik değer ünitesi için de benzer hesaplamalar (6.13.) ve (6.14.) formülleri ile yapılır.

$$D_{jk}^{a}(t) = D_{jk}^{a}(t-1) + \Delta D_{jk}^{a}(t) \quad (6.12.)$$

$$\Delta\beta_{k}^{c}(t) = \lambda\delta_{k} + \alpha\Delta\beta_{k}^{c}(t-1) \quad (6.13.)$$

$$\beta_{k}^{c} = \beta_{k}^{c}(t-1) + \Delta\beta_{k}^{c}(t) \quad (6.14.)$$

O_{l}^{i}, i. işlem elemanının çıktısını göstermek üzere, gizli katman-girdi katmanı arasındaki ağırlıkların değiştirilmesi ise benzer şekilde (6.15.) eşitliği ile sağlanır.

$$\Delta D_{lj}^{i}(t) = \lambda\delta_{j}^{a}O_{l}^{i} + \alpha\Delta D_{lj}^{i}(t-1) \quad (6.15.)$$

Buradaki hata terimi ise (6.16.) eşitliği ile hesaplanacaktır.

$$\delta_{j}^{a} = f'(NET)\sum_{k}\delta_{k}D_{jk}^{a} \quad (6.16.)$$

Aktivasyon fonksiyonu sigmoid fonksiyon olduğu için (6.16.) eşitliği, (6.17.) eşitliği ile ifade edilebilir.

$$\delta_{j}^{a} = C_{j}^{a}(1 - C_{j}^{a})\sum_{m}\delta_{k}D_{jk}^{a} \quad (6.17.)$$

Bu durumda ağırlıkların yeni değeri (6.18.) eşitliği ile hesaplanacaktır.

$$D_{lj}^{i}(t) = D_{lj}^{i}(t-1) + \Delta\mathrm{D}_{\mathrm{lj}}^{\mathrm{i}}(\mathrm{t}) \quad (6.18.)$$

Burada bulunan eşik değer ünitesinin yeni ağırlıkları da (6.19.) ve (6.20.) formülleri ile hesaplanır.

$$\Delta\beta_{j}^{a}(t) = \lambda\delta_{j}^{a} + \alpha\Delta\beta_{j}^{a}(t-1) \quad (6.19)$$

$$\beta_{j}^{a}(t) = \beta_{j}^{a}(t-1) + \Delta\beta_{j}^{a}(t) \quad (6.20.)$$

<u>ADIM 6:</u> Durdurma kriteri:

Yukarıda verilen adımlar deney tasarımında belirlenen iterasyon sayısı tamamlanana kadar tekrarlanarak, hata en küçüklenmeye çalışılır ve hatayı en küçükleyen ağırlık kümeleri kural çıkarımı için saklanır.

6.5.2. TAKKO Algoritması ile Kural Üretimi

TAKKOYSA-sınıflandırıcısı ile yapay sinir ağlarında gizli olan bilgiyi doğru ve anlaşılır sınıflandırma kurallarına dönüştürmek için, tur atan karınca koloni optimizasyonu temelli bir kural üretim modülü tasarlanmıştır. Bu modül, çok katmanlı algılayıcılardan gelen ağırlık ve ikili veri kümeleri ile çalışmaktadır. Aşağıda geliştirilen algoritmanın TAKKO ile kural üretim modülü açıklanmıştır.

Problem formülasyonu:

ÇKA, bir gizli katman kullanılarak eğitildiğinde iki grup ağırlık elde edilir. Birinci grup (w_1) girdi katmanı (i)'den gizli katman (j)'ye ağırlıkları ve ikinci grup (w_2), gizli katman (j)'den çıktı katmanı (k)'ya ağırlıkları içerir. j. gizli katman işlem elemanına toplam girdi GGK_j, (6.21.) eşitliği ile ifade edilir.

$$GGK_j = \sum_{i=1}^{I} x_i w1_{ij} \tag{6.21.}$$

aynı işlem elemanının çıktısı CGK_j, sigmoid aktivasyon fonksiyon kullanıldığı için (6.22.) eşitliği ile hesaplanır;

$$CGK_j = \frac{1}{1+e^{-GGK_j}} \tag{6.22.}$$

k. çıktı düğümün toplam girdisi $GÇK_k$ (6.23.) ise eşitliği ile hesaplanır;

$$GCK_k = \sum_{j=1}^{J} w2_{jk}.CGK_j \tag{6.23.}$$

Sonuç olarak, *k*. çıktı düğümün son değeri yani çıktısı ise (6.24.) eşitliği kullanılarak hesaplanır;

$$Ç_k = \frac{1}{1+e^{-GCK_k}} \tag{6.24.}$$

Eğer ÇKA bir yerine iki gizli katman kullanılarak eğitilirse, bu durumda üç grup ağırlık elde edilir;

w_1: girdi katmanı (i)'den gizli katman (j)'ye ağırlıklar,
w_2: gizli katman (j)'den gizli katman (m)'e ağırlıklar,
w_3: gizli katman (m)'den çıktı katmanı (k)'ya ağırlıklar

Bu ağırlıklar kullanılarak, k. çıktı düğümün son değeri $Ç_k$, (6.25.) ve (6.26.) eşitlikleri kullanılarak hesaplanır;

$$Ç_k = \frac{1}{1+e^{-GCK_k}} \tag{6.24.}$$

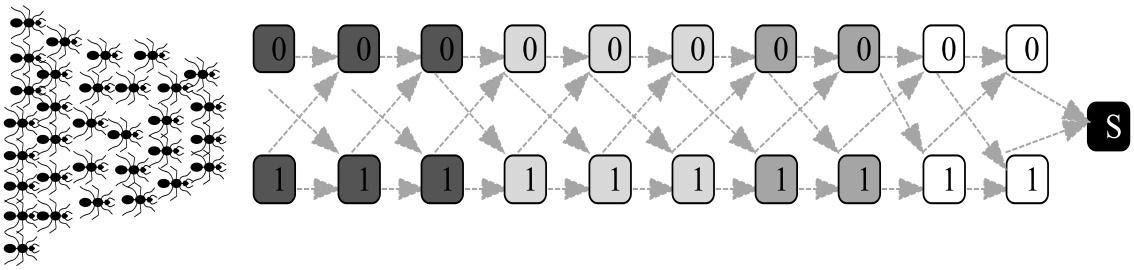
$$GCK_k = \sum_{m=1}^{M} w3_{mk} \cdot \left(\frac{1}{1+e^{-\sum_{j=1}^{J} w2_{jm} \cdot \left(\frac{1}{1+e^{\sum_{i=1}^{I} x_i \cdot w1_{ij}}} \right)}} \right) \tag{6.25.}$$

$Ç_k$ fonksiyonu üstel bir fonksiyondur ve maksimum çıktı değeri 1'dir. Sonuç olarak girdi nitelikleri ile ilgili sınıflar arasında kurallar çıkarmak için $Ç_k$'yı en büyükleyen girdi vektörünün bulunması gerekir. Bu bir optimizasyon problemidir ve şu şekilde gösterilebilir [4, 149];

$$En\ büyükle \quad C_K$$
$$x_i = 0\ veya\ 1 \tag{6.26.}$$

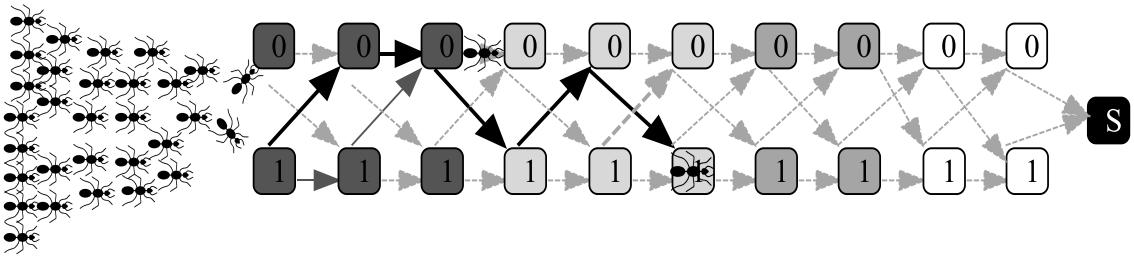
Kodlama ve başlangıç popülasyonu:

Kural gösteriminde daha önce bahsedilen ikili yapı kullanılmıştır. Şekil 6.3. "Tenis Oynama" veri kümesi için geliştirilen algoritmada karıncaların kullandığı ikili yolları göstermektedir. Şekilde henüz hiçbir karınca arama işlemine başlamamıştır ve yolların feremon ve sıklık miktarları başlangıç değerlerindedir.



Şekil 6.3. Geliştirilen algoritmada karıncaların kullandığı ikili yollara örnek.

TAKKO tabanlı kural çıkarımında başlangıç popülasyonu rastgele üretilmektedir. Şekil 6.4. yapay yolların karıncalar tarafından rastgele üretimini göstermektedir. Geliştirilen algoritma her sınıf için ayrı çalışmaktadır. Dolayısıyla her bir sınıf için belirlenen sayıda karınca, belirlenen iterasyon sayısı boyunca yapay yollar kat ederek ilgili sınıfa ait kuralları üretmektedir.



Şekil 6.4. Karıncalar tarafından yapay yolların rastgele üretilmesi.

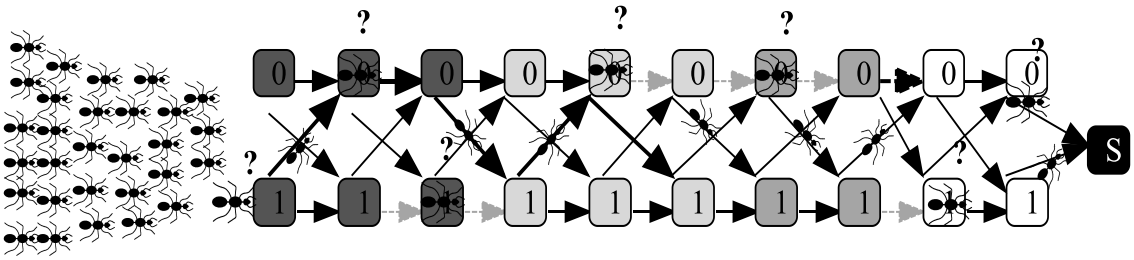
Yapay karıncalar için yapay yolların üretilmesi:

TAKKO algoritmasında Bölüm 5'de bahsedildiği gibi yapay yollar bir olasılık fonksiyonuna göre üretilir. Ancak temel tur atan karınca koloni optimizasyonu algoritması, sadece feremon tabanlı yön seçim stratejisi kullanmasından dolayı erken yakınsama gibi bir dezavantaja sahiptir. Bu dezavantajın üstesinden gelmek için tabu arama algoritmasının sıklık tabanlı hafıza özelliği kullanılabilir [177]. TAKKO'da sıklık tabanlı hafıza bir alt yolun karıncalar tarafından ne sıklıkla kullanıldığını tutmaktadır. Eğer karıncalar bazen yollarını bu prensibe göre seçerlerse, farklı yollar izleyebilirler, yerel optimuma takılma ihtimalleri azalır ve genel optimumu yakalayabilirler.

Geliştirilen algoritmada 0 ve 1 $(0 \to 1)$ bitleri arasındaki alt yolun tercih edilmesi olasılığının hesaplanmasında temel TAKKO algoritmasından farklı olarak (6.27.) eşitliği kullanılmıştır. Burada f, sıklık faktörüdür $(f \geq 1)$. Eğer $(f * f_{01} < f_{00})$ şartı sağlanırsa, bu durumda $(0 \to 1)$ yolu doğrudan seçilir; aksi halde feremon tabanlı yol seçimi stratejisi kullanılır.

$$p_{01}(t) = \begin{cases} 1 & eğer\ (f * f_{01} < f_{00} \\ \frac{\tau_{01}}{\tau_{01}+\tau_{00}} & aksi\ halde \end{cases} \qquad (6.27.)$$

Şekil 6.5.'de tenis oynama veri kümesi ikili gösteriminde karıncaların (6.27.) eşitliğinde verilen olasılık fonksiyonuna göre yol seçim stratejisini gösterilmektedir. Her bir karınca olasılık fonksiyonunu kullanarak bulunduğu noktadan 0 noktasına mı yoksa 1 noktasına mı ilerleyeceğine karar vermektedir. Şekilde noktalar arasındaki çizgilerin kalınlıkları ve ton farklılıkları önceki iterasyonlarda karıncalar tarafından daha yoğun kullanılan yolların feremon ve sıklık miktarlarının artmış olduğunu ifade etmektedir.



Şekil 6.5. Karıncalar tarafından olasılık fonksiyonuna göre yapay yolların seçilmesi.

Feremon ve sıklık miktarlarının güncellenmesi:

Yapay feremon ve sıklık sırasıyla (6.28.) ve (6.29.) eşitlikleri kullanılarak hesaplanır. Eşitlik (6.28)'de $\Delta\tau_{01}^{k}$ k. yapay karınca tarafından $(0 \to 1)$ alt yoluna yapıştırılan feremon maddesi miktarıdır; Q pozitif sabittir ve F_k, karınca tarafından bulunan çözüm kullanılarak hesaplanan amaç fonksiyon değeridir. Önerilen algoritmada amaç fonksiyon değeri olarak yapay sinir ağının çıktı fonksiyon değeri yani, kalite değeri C_k kullanılmıştır.

$$\Delta\tau_{01}^{k}(t,t+1)=\begin{cases}Q*F_k & eğer\ k\ karıncası\ alt\ yol\ (0-1)'den\ geçerse\\ 0 & aksi\ halde\end{cases} \quad (6.28.)$$

$$\Delta F_{ij}^{k}(t,t+1)=\begin{cases}1 & eğer\ i-j\ yolu\ kullanılırsa\\ 0 & aksi\ durumda\end{cases} \quad (6.29.)$$

Bütün karıncalar araştırma işlemini tamamladıktan ve yolları belirledikten sonra $(0\rightarrow 1)$ alt yoluna, *t* ve (*t+1*) zaman aralığında atanacak feremon maddesi $\Delta\tau_{01}(t,t+1)$ ve sıklık miktarları $\Delta F_{01}(t,t+1)$, (6.30.) ve (6.31.) eşitlikleri kullanılarak hesaplanır.

$$\Delta\tau_{01}(t,t+1)=\sum_{k=1}^{M}\Delta\tau_{01}^{k}(t,t+1) \quad (6.30.)$$

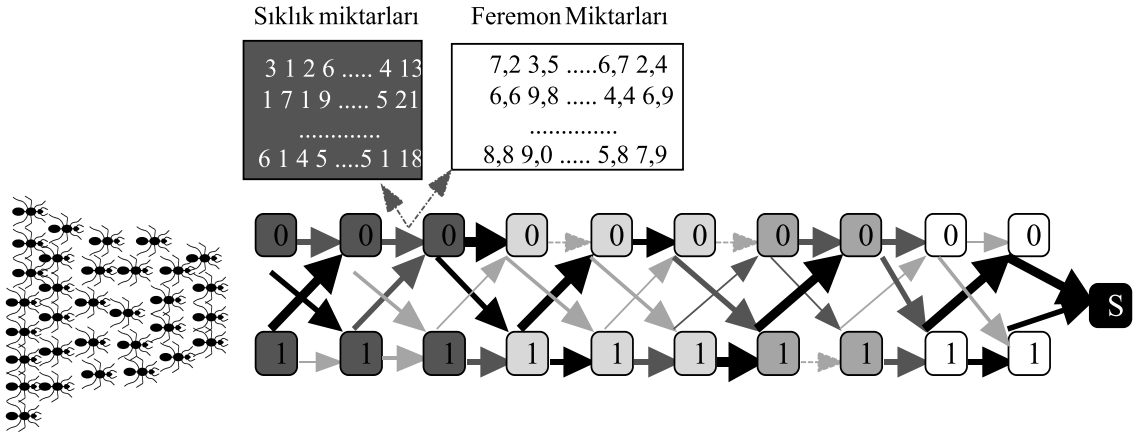
$$\Delta F_{01}(t,t+1)=\sum_{k=1}^{M}\Delta F_{01}^{k}(t,t+1) \quad (6.31.)$$

Aynı alt yolun (*t+1*) anındaki feremon ve sıklık miktarları ise (6.32.) ve (6.33.) eşitlikleri ile belirlenir. Burada ρ, buharlaşma parametresi olarak adlandırılır ve $(1-\rho)$, feremon maddesinin buharlaşma miktarını göstermektedir $(0\le\rho<1)$.

$$\tau_{01}(t+1)=\rho\tau_{01}(t)+\Delta\tau_{01}(t,t+1) \quad (6.32.)$$

$$F_{01}(t+1)=F_{01}(t)+\Delta F_{01}(t,t+1) \quad (6.33.)$$

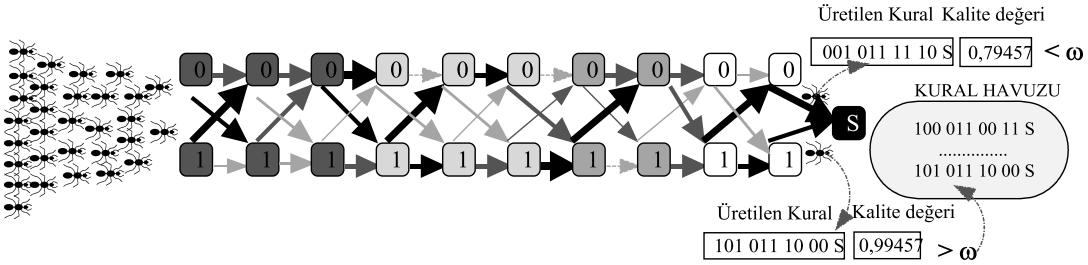
Şekil 6.6. tüm karıncalar yollarını tamamladıktan sonra, oluşan alt-yollara yapıştırılan feremon ve sıklık miktarlarını göstermektedir. Görüldüğü gibi kullanımına bağlı olarak, her yolun sıklık ve feremon miktarları farklıdır. Geliştirilen algoritmada feremon ve sıklık miktarları çok boyutlu dizilerde tutulmaktadır. Şekilde örnek sıklık ve feremon dizisi verilmektedir.



Şekil 6.6. Feremon ve sıklık güncelleme.

Aday kuralların belirlenmesi:

Kural indirgeme prosedürü için aday çözümler (kurallar) bu aşamada belirlenir. Amaç yapay sinir ağını dilsel olarak ifade etmek olduğu ve aktivasyon fonksiyonu olarak da 0-1 aralığında değer üreten sigmoid fonksiyon kullanıldığı için; kural indirgemeye aday olarak kalite değeri (C_k) 1'e yakın olan çözümler saklanır. Kalite değeri 1'e yakın olan kurallar, geliştirilen algoritmada, C_k değeri belirli bir "ω" değerin üstünde kalan kurallardır. Şekil 6.7.'de aday ve aday olmayan kurala örnek gösterilmektedir.



Şekil 6.7. Aday kural örneği.

Durdurma kriteri:

TAKKO tabanlı kural üretim algoritması belirli bir iterasyon sayısı sağlanana kadar devam ettirilir. Daha sonra veri kümesini en iyi yansıtacak kural kümesinin belirlenmesi için kural indirgeme prosedürü gerçekleştirilir.

6.5.3. Kural İndirgeme

Kural indirgeme sürecinde, kurallar çözüm dizilerinin uygunluk değerine göre seçilir. Uygunluk değeri, elde edilen kuralların veri kümesini temsil etme gücünü ortaya koymaktadır.

Bir kural verilen eğitim örneğini sınıflandırmakta kullanıldığında; pozitif doğru (tp), pozitif yanlış (fp), negatif doğru (tn) ve negatif yanlış (fn) olmak üzere dört durum ortaya çıkar [3]. Pozitif doğru ve negatif doğru, doğru sınıflandırmaları; pozitif yanlış ve negatif yanlış, yanlış sınıflandırmaları ifade eder.

- *Pozitif doğru (tp):* Kural sınıfın pozitif olduğunu tahmin eder ve verilen örneğin sınıfı da pozitiftir.
- *Negatif doğru (tn):* Kural sınıfın negatif olduğunu tahmin eder ve verilen örneğin sınıfı da negatiftir.
- *Pozitif yanlış (fp):* Kural sınıfın pozitif olduğunu tahmin eder fakat verilen örneğin sınıfı negatiftir.
- *Negatif yanlış (fn):* Kural sınıfın negatif olduğunu tahmin eder fakat verilen örneğin sınıfı pozitiftir.

Bu durumların örnek üzerinde gösterimi "Tenis Oynama" veri kümesi için Tablo 6.6.-6.9.'da verilmektedir. Tablolarda ilk satır verilen örneği, ikinci satır ise üretilen kuralı ifade etmektedir.

Tablo 6.6. Pozitif doğru durumu.

	Hava Durumu			Sıcaklık			Nem		Rüzgar		Karar	
	Güneşli a_1	Bulutlu a_2	Yağmurlu a_3	Sıcak a_4	Ilık a_5	Soğuk a_6	Yüksek a_7	Normal a_8	Hafif a_9	Sert a_{10}	Oynama C_1	Oyna C_2
örnek	**1**	0	0	**1**	0	0	**1**	0	**1**	0	**1**	0
kural	1	0	1	1	0	0	1	0	1	1	1	0

Tablo 6.7. Negatif doğru durumu.

	Hava Durumu			Sıcaklık			Nem		Rüzgar		Karar	
	Güneşli a_1	Bulutlu a_2	Yağmurlu a_3	Sıcak a_4	Ilık a_5	Soğuk a_6	Yüksek a_7	Normal a_8	Hafif a_9	Sert a_{10}	Oynama C_1	Oyna C_2
örnek	**1**	0	0	**1**	0	0	**1**	0	**1**	0	**1**	0
kural	**0**	1	0	1	1	0	**0**	1	**1**	0	**0**	1

Tablo 6.8. Pozitif yanlış durumu.

	Hava Durumu			Sıcaklık			Nem		Rüzgar		Karar	
	Güneşli a_1	Bulutlu a_2	Yağmurlu a_3	Sıcak a_4	Ilık a_5	Soğuk a_6	Yüksek a_7	Normal a_8	Hafif a_9	Sert a_{10}	Oynama C_1	Oyna C_2
örnek	**1**	0	0	**1**	0	0	**1**	0	**1**	0	**1**	0
kural	1	0	1	1	1	0	1	0	1	0	0	1

Tablo 6.9. Negatif yanlış durumu.

	Hava Durumu			Sıcaklık			Nem		Rüzgar		Karar	
	Güneşli a_1	Bulutlu a_2	Yağmurlu a_3	Sıcak a_4	Ilık a_5	Soğuk a_6	Yüksek a_7	Normal a_8	Hafif a_9	Sert a_{10}	Oynama C_1	Oyna C_2
örnek	**1**	0	0	**1**	0	0	**1**	0	**1**	0	**1**	0
kural	**0**	0	1	**1**	1	0	**0**	1	**1**	0	**1**	0

Kuralın verilen örneği sağlayıp sağlamadığının kontrolü her bir değişkenin alt-dizi bitlerine bakılarak yapılmaktadır. Veri gösterimi bölümünde anlatıldığı gibi veride her bir değişken alt-dizi bitlerinin yalnız bir tanesi "1" değerini alabilmektedir. Dolayısıyla örnek ile aynı bit değerleri "1" olan veya ele alınan değişken ile ilgili tüm alt-dizi bitleri aynı olan (tüm alt-dizi bit değerleri 0 veya 1 olan) kuralların önde gelen kısımları örtüşür. Eğer sınıf alt-dizi bölümlerinde de "1" değerini alan bitler örtüşürse bu durumda Tablo 6.6.'da olduğu gibi kural "pozitif doğru" durumunu yansıtır. Eğer kuralın önde gelen kısmı örtüşür fakat Tablo 6.8.'de olduğu gibi sınıf kısımları örtüşmezse, kural "pozitif yanlış" durumu ifade eder.

Tablo 6.7.'de olduğu gibi kuralın önde gelen kısmında örtüşme olmaz (mesela tabloda hava durumu değişkeni örnekte güneşli değerini alırken, kuralda yağmurlu değerini almış; yani kural ile örnek farklı durumları ifade ediyor) ve sınıf bölümleri de tutarsızsa bu durumda kural, negatif doğru durumunu yansıtır. Eğer Tablo 6.9.'da olduğu gibi eğer

kuralın önde gelen kısmı örnek ile örtüşmez fakat sınıf kısmı örtüşürse bu durumda bu kural negatif yanlış durumunu ortaya koyar. Dört durum ile ilgili kıyaslama sonuçları Tablo 6.10.'da gösterilmektedir.

Tablo 6.10. Durumsallık tablosu.

Önde gelen bölümlerde örtüşme	*Sınıf bölümünde örtüşme*	*Durum*
√	√	tp (pozitif doğru)
×	×	tn (negatif doğru)
√	×	fp (pozitif yanlış)
×	√	fn (negatif yanlış)

Bir kuralın önde gelen kısmındaki tüm bit değerleri aynı değeri alırsa bu durumda bu kural anlamsız bir kuraldır, yani hiçbir durumu ifade etmez. Bu kurallar için karşılaştırma ve hesaplama yapmaya gerek yoktur. Geliştirilen algoritma için bu tip kurallar anlamsız kurallardır ve değerlendirmeye alınmaz.

Bu durumlar kullanılarak hassaslık ve belirlilik ölçütleri ifade edilebilir. Hassaslık (*Se*) doğru sınıflandırılan gerçek pozitif örneklerin kesitini ölçerken, belirlilik (S_p) doğru sınıflandırılan gerçek negatif örneklerin kesitini ölçer [90]. Bu kavramlar kullanılarak, uygunluk değeri (6.35) fonksiyonu ile hesaplanır [11];

$$S_e = \frac{tp}{tp+fn} \qquad S_p = \frac{tn}{tn+fp} \tag{6.34.}$$

$$Uygunluk = S_e \times S_p \tag{6.35.}$$

Kural havuzunda bulunan aday kuralların uygunluk değerlerine göre kurallar indirgenir. Uygunluk değeri en yüksek olan kuraldan başlanarak, eğitim veri kümesinde sınıflandırılmamış hiçbir örnek veya kural havuzunda değerlendirilecek kural kalmayana kadar bir kural listesi oluşturulur. Bu işlem sırasında değerlendirmeye alınan kuralın sağladığı durumlar eğitim kümesinden çıkarılır. Çünkü buradaki temel amaç, ilgili probleme ait veri kümesinin en az sayıda kural ile ifade edilmesidir. Uygunluk değeri en yüksek olan kuraldan indirgeme işlemine başlanmasının da nedeni budur, çünkü bu kural veri kümesindeki daha çok örneği ifade edebilir.

Kural listesi belirlendikten sonra, bu listenin eğitim doğruluğu ve kural sayısı hesaplanır. Doğruluk, sınıflandırıcının doğru sonuçlar üretmekteki yeteneğini ölçen bir ölçüttür ve (6.36.) eşitliği ile hesaplanır;

$$Doğruluk = \frac{Kural\ listesinin\ doğru\ sınıflandırdığı\ örnek\ sayısı}{Veri\ kümesindeki\ örnek\ sayısı} \quad (6.36.)$$

(6.36.) eşitliğinde paydada eğitim veri kümesi ele alınırsa ve bu küme üzerinde doğru sınıflandırılan örnek belirlenirse eğitim doğruluğu hesaplanmış olur. Eğer test veri kümesi ele alınırsa bu durumda test doğruluğu hesaplanır. Geliştirilen algoritmanın tahminleyici doğruluğu test doğruluğu ile belirlenmiştir.

6.5.4. Kural Basitleştirme ve Dilsel Kurallara Dönüştürme

Yapay sinir ağlarından kural çıkarımındaki temel amaç, YSA'nın içindeki gizli bilginin kullanıcının anlayacağı dilsel kurallara dönüştürülmesi olduğu için oluşturulan kural listesi basitleştirilerek dilsel kurallara dönüştürülür. Bu aşamada izlenen yol şu şekilde açıklanabilir;

- Kuraldaki bir değişkene ait tüm bit değerleri aynı değeri içeriyorsa (hepsi 0 veya hepsi 1) bu durumda bu değişken dilsel kuralda yer almaz ve kuraldan çıkarılır. Bu işlem sayesinde daha basit kurallar elde edilebilir (*basitleştirme)*. Bu şekilde olası tüm değerleri alabilen bir değişkenin kuralda yer almasının veya almamasının kuralı etkilemeyeceği açıktır.
- Kuraldaki bir değişkene ait birden fazla bit değeri "1" değerini almışsa ve değişkene ait bitlerden en az biri farklı değere sahipse bu durumda "1" değerini içeren bitlerdeki dilsel değişkenler mantıksal "VEYA" operatörü ile birleştirilir.
- İki farklı değişken dilsel kuralda birleştirilirken mantıksal "VE" operatörü kullanılır.

Tablo 6.11. Tenis Oynama veri kümesi üzerinde çıkarılan bir ikili kuralın dilsel kurala dönüştürülmesi ve basitleştirilmesi işlemini göstermektedir.

Tablo 6.11. Kural basitleştirme ve dilsel kurallara dönüştürme örneği.

Hava Durumu, m_1=3			Sıcaklık, m_2=3			Nem,m_3=2		Rüzgar,m_4=2		Tenis Oyna	
Güneşli	Bulutlu	Yağmurlu	Sıcak	Ilık	Soğuk	Yüksek	Normal	Hafif	Sert	Hayır	Evet
a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	C_1	C_2
1	0	1	1	0	1	0	0	1	1	1	0
Dilsel kural: **EĞER** hava durumu güneşli **VEYA** yağmurlu **VE** sıcaklık sıcak **VEYA** soğuk **VE** rüzgar hafif veya sert ise **O HALDE** tenis oynama.											
Basitleştirilmiş dilsel kural: EĞER hava durumu güneşli **VEYA** yağmurlu **VE** sıcaklık sıcak **VEYA** soğuk ise **O HALDE** tenis oynama.											

Tablodan da görüldüğü gibi rüzgar değişkeni {hafif, sert} olan olası iki değerini de almaktadır. Dolayısıyla bu değişken kuralda bir anlam ifade etmemektedir. Benzer şekilde nem değişkeni de olası {yüksek, normal} değerlerinin her ikisini de almamaktadır. Bu değişkenin kuralda yer alması da anlamsızdır. Rüzgar ve nem değişkenlerinin bit değerleri aynı değeri aldığı için bu değişkenler kuralda yer almayacak ve çıkarılacaktır.

Hava durumu değişkeni ise olası {güneşli, bulutlu, yağmurlu} değerlerini ifade eden üç bit değerinden ikisini almaktadır. Dolayısıyla bit değerlerinde "1" içeren değerler "VEYA" mantıksal operatörü kullanılarak "güneşli VEYA yağmurlu" şeklinde birleştirilecektir. Aynı işlem sıcaklık değişkeni için de yapılacaktır.

Son olarak basitleştirme işleminden sonra kalan değişkenlerin dilsel kuralda birleştirilmesinde "VE" operatörü kullanılarak Tablo 6.11.'in son satırında yer alan basitleştirilmiş dilsel kural elde edilecektir.

6.6. Sonuç

Bu bölümde, YSA'larında gizli olan bilgiyi kullanıcının anlayacağı dilsel kurallara dönüştürmek için geliştirilen algoritma anlatılmıştır. Geliştirilen algoritma eğitilmiş yapay sinir ağlarından sınıflandırma kuralları çıkarmak için aktivasyon fonksiyonu üzerine herhangi bir tahmin yapmamakta, sadece ağırlık değerlerini kullanmaktadır.

Tur atan karınca koloni optimizasyonu algoritması, elde edilen ağırlıkları kullanarak yapay sinir ağının çıktı fonksiyonunu en büyükleyen kuralların üretilmesinde

kullanılmıştır. TAKKO ile üretilen kurallar indirgenerek dilsel kurallara dönüştürülmüştür. TAKKOYSA-sınıflandırıcısı ile elde edilen sonuçlar Bölüm 7'de verilmiştir.

7. BÖLÜM
DENEYSEL ÇALIŞMA VE ANALİZLER

7.1. Giriş

Çalışmanın bu bölümünde geliştirilen TAKKOYSA-sınıflandırıcısının elde ettiği sonuçlar çeşitli kriterler açısından değerlendirilmiştir. Değerlendirmede UCI (University of California at Irvine) makine öğrenme deposundan [178] alınan 12 farklı ikili ve çok sınıflı referans veri kümesi kullanılmıştır. Test problemleri üzerinde Taguchi yöntemi ile deney tasarımı gerçekleştirilerek algoritma faktörleri performans ölçütü olan test doğruluklarına göre sıralanmıştır. Faktörlerin etkinlikleri analiz edilmiş ve her bir faktörün alması gereken düzeyler belirlenmiştir.

Belirlenen faktör düzeyleri ile geliştirilen algoritmanın elde ettiği sonuçlar ele alınan etkinlik ölçütleri bakımından incelenmiştir. Son bölümde algoritmanın performansı iki aşamada değerlendirilmiştir. İlk aşamada geliştirilen algoritma dört farklı klasik makine öğrenme algoritması ile karşılaştırılmış ve istatistiksel analizler gerçekleştirilmiştir. İkinci aşamada ise algoritmanın elde ettiği sonuçlar literatürde mevcut olan kural tabanlı sınıflandırıcılar ile karşılaştırılmıştır. Bu karşılaştırmalara yapay sinir ağlarından kural çıkaran algoritmalar da dahil edilmiştir.

7.2. Veri Kümeleri

UCI makine öğrenme deposundan alınan 12 farklı veri kümesi geliştirilen algoritmanın performansını test etmek için kullanılmıştır. Veri kümelerinin özellikleri Tablo 7.1.'de gösterilmektedir.

Tablo 7.1. Veri kümelerinin temel özellikleri.

Veri Kümesi	Örnek Sayısı	Kategorik Değişken Sayısı	Sürekli Değişken Sayısı	Boş Değişken Değeri	Sınıf Sayısı
CRX	690	9	6	Evet	2
Heart-C	303	8	5	Evet	5
Iris	150	-	4	Hayır	3
LBC	286	9	-	Evet	2
Monks-2	432	6	-	Hayır	2
Nursery	12 960	8	-	Hayır	5
Pima	768	-	8	Hayır	2
Spect-heart	267	22	-	Hayır	2
Tic-Tac-Toe	958	9	-	Hayır	2
Vote	435	16	-	Evet	2
WBC	699	-	9	Evet	2
Zoo	101	17	-	Hayır	7

Bu veri kümeleri arasından sekizi ikili, kalan dördü çoklu sınıflandırma problemleridir. Bazı veri kümeleri eksik değerler içermektedir fakat bu örnekler veri kümelerinden çıkarılmamış sadece eksikliğin olduğu parametre değeri çıkartılmıştır.

Bölüm 6'da da değinildiği gibi geliştirilen TAKKOYSA-sınıflandırıcısı, ikili değişkenler üzerinde çalışmaktadır. Kesikli değişkenler doğrudan Bölüm 6'da anlatıldığı gibi ikili alt-dizilere kodlanabilmektedir. Ancak sürekli değişkenlerin ikili forma dönüştürülmeden önce kesiklendirilmesi gerekmektedir. Kesiklendirme işlemi, Weka 3.5 yazılımı kullanılarak "Basit Bölme -Simple Binning" ile gerçekleştirilmiştir. Basit bölme yöntemi, ele alınan sürekli değişkeni, ayrılacağı aralık sayısına göre "eşit genişlik"te parçalara bölmektedir. Örnek olarak (0-100) arası değer alan bir sayısal değişken ele alındığında ve bu değişkenin dört aralığa bölünmesi istendiğinde basit bölme algoritmasının oluşturacağı kesikli aralıklar şu şekilde olacaktır: {(0, 25), (25, 50), (50, 75), (75,100)}. Basit bölme algoritmasında "eşit genişlik yerine", "eşit frekans" ölçütü de kullanılabilir. Bu durumda algoritma, her bir kesikli aralığa eşit miktarda örnek atayacaktır. Tez çalışmasında basit bölme algoritmasının "eşit genişlik" ölçütü kullanılmıştır.

Monks-2, Spect-heart ve Zoo veri kümeleri dışında kalan veri kümelerinde tahminleyici doğruluk çok iyi bilinen bir yöntem olan çapraz-geçerlilik testi ile ölçülmüştür. Çapraz-geçerlilik testinde 10-katlı çapraz doğrulama kullanılmıştır. Her veri kümesi 10 alt veri kümesine bölünmüş ve geliştirilen kural çakırım algoritması 10 kez çalıştırılmıştır. Her bir çalıştırmada farklı bir alt-veri kümesi test veri kümesi, kalan 9 parça ise eğitim kümesi olarak kullanılmıştır. Test veri kümesi üzerindeki tahminleyici doğruluk on farklı çalıştırmanın ortalamaları alınarak hesaplanmıştır.

Monks-2 ve Spect-heart veri kümelerine 10-katlı çapraz geçerlilik testinin uygulanmamasının nedeni, bu veri kümelerinin orjinalinde tek bir eğitim ve test kümelerine ayrılmasıdır. Zoo veri kümesinin ise az sayıda örnek ve çok sayıda sınıf içermesinden dolayı bu veri kümesine de 10-katlı çapraz geçerlilik testi uygulanmamıştır. Veri kümesinin % 67'si eğtitim kalan % 33'ü ise test veri kümesi olarak ele alınmıştır.

***Credit Approval veri kümesi (CRX)*:** CRX veri kümesi kredi kartı uygulamaları ile ilgili verileri içermektedir. Veri kümesinin sürekli, küçük değerli kesikli ve büyük değerli kesikli değişkenlerin iyi bir karışımını içermesinden dolayı bu veri kümesi geliştirilen algoritmanın performansını değerlendirmek için kullanılmıştır. CRX veri kümesinde 690 örnek, 9 kategorik ve 6 sürekli değişken bulunmaktadır. 37 örnek bir veya daha fazla boş değer içermektedir. Sınıf değişkeninin % 44.5'i '+' sınıfına, % 55.5'i '-' sınıfına aittir. Tablo 7.2. CRX veri kümesi değişkenleri hakkında bilgi vermektedir. Tablonun 2. sütunu orijinal veri ile ilgili bilgiyi 3. ve 4. sütunu ise geliştirilen algoritmada kullanılan ve kesiklendirme işleminden sonra elde edilen ikili değişkenler ile ilgili bilgileri içermektedir. Tablodan da görülebileceği gibi kesiklendirme işleminden sonra toplam 54 ikili girdi değişkeni elde edilmiştir.

Tablo 7.2. CRX veri kümesi değişkenleri.

Değişken	Değişken Bilgisi	İkili Değişken Sayısı	Alt-Aralık
A1	b, a	2	{b}, {a}
A2	Sürekli	2	[-; 38,96), [38,96; -)
A3	Sürekli	2	[-; 4,2075), [4,2075; -)
A4	u, y, l, t	4	{u}, {y}, {l}, {t}
A5	g, p, gg	3	{g}, {p}, {gg}
A6	c, d, cc, i, j, k, m, r, q, w, x, e, aa, ff	14	{c}, {d}, {cc}, {i}, {j}, {k}, {m}, {r}, {q}, {w}, {x}, {e}, {aa}, {ff}
A7	v, h, bb, j, n, z, dd, ff, o	9	{v}, {h}, {bb}, {j}, {n}, {z}, {dd}, {ff}, {o}
A8	Sürekli	2	[-; 1,02), [1,02; -)
A9	t, f	2	{t}, {f}
A10	t, f	2	{t}, {f}
A11	Sürekli	3	[-; 0,5), [0,5; 2,5),[2,5; -)
A12	t, f	2	{t}, {f}
A13	g, p, s	3	{g}, {p}, {s}
A14	Sürekli	2	[-; 105), [105; -)
A15	Sürekli	2	[-; 492), [492; -)
A16 (Sınıf)	+, -	2	

***Heart-C veri kümesi*:** Heart-C veri kümesi dört farklı tıbbi merkezden toplanan kalp hastalıkları verilerini içermektedir. Veri kümesinde 303 örnek ve 76 değişken bulunmaktadır. Ancak yayınlanan bütün deneysel çalışmalar, bu 76 değişkenin sadece biri sınıf değişkeni olmak üzere, 14'ünün kullanımını önermektedir. Heart-C veri kümesinin sınıf değişkenini, örneğin ait olduğu hastada kalp hastalığının olup olmadığı bilgisi oluşturmaktadır. Sınıf değişkeni kalp hastalığının ağırlığını yansıtan {0, 1, 2, 3, 4} değerlerini almaktadır. Veri kümesi eksik değerler içermektedir. Sınıf değişkeninin % 54'ü '0' sınıfına, % 18'i '1' sınıfına, % 12'si '2' sınıfına, % 12'si '3' sınıfına ve % 4'ü '4' sınıfına aittir. Tablo 7.3. Heart-C veri kümesi değişkenlerine ilişkin bilgileri içermektedir. Tablodan da görülebileceği gibi kesiklendirme işleminden sonra toplam 38 ikili girdi değişkeni elde edilmiştir.

Tablo 7.3. Heart-C veri kümesi değişkenleri.

Değişken	Değişken Bilgisi	İkili Değişken Sayısı	Alt-Aralık
Age	Sürekli	3	[-; 45), [45; 61), [61; -)
Sex	Erkek, Kadın	2	{erkek}, {kadın}
Cp	1, 2, 3, 4	4	{1}, {2}, {3}, {4}
Trestbps	Sürekli	3	[-; 129.33), [129.33; 164.67), [164.67; -)
Chol	Sürekli	3	[-; 272), [272; 418), [418; -)
Fbs	Doğru, yanlış	2	{doğru}, {yanlış}
Restecg	0, 1, 2	3	{0}, {1}, {2}
Thalach	Sürekli	3	[-; 114.67), [114.67; 158.33), [158.33; -)
Exang	Evet, hayır	2	{evet}, {hayır}
Oldpeak	Sürekli	3	[-; 2.07), [2.07; 4.13), [4.13; -)
Slope	1, 2, 3	3	{1}, {2}, {3}
Ca	0, 1, 2, 3	4	{0}, {1}, {2}, {3}
Thal	3, 6, 7	3	{3}, {6}, {7}
Num (Sınıf)	0, 1, 2, 3, 4	5	

Iris veri kümesi: Iris, makine öğrenme literatüründe en çok bilinen ve kullanılan veri kümesidir. Bu veri kümesinde her biri 50 örnek içeren, iris bitkisi türüyle ilgili 3 sınıf bulunmaktadır. Ayrıca veri kümesi 4 sürekli değişken içermektedir. Kesiklendirme ve ikili forma dönüştürme işleminden sonra 12 ikili girdi değişkeni elde edilmiştir. İlgili girdilere ait bilgiler Tablo 7.4.'de yer almaktadır.

Tablo 7.4. Iris veri kümesi değişkenleri

Değişken	Değişken Bilgisi	İkili Değişken Sayısı	Alt-Aralık
Sepal length	Sürekli	3	[-; 5.55], [5.555; 6.15),[6.15; -)
Sepal width	Sürekli	3	[-; 2.95), [2.95; 3.35), [3.35; -)
Petal length	Sürekli	3	[-; 2.45), [2.45; 4.75), [4.75; -)
Petal width	Sürekli	3	[-; 0.8), [0.8; 1.75), [1.75; -)
Sınıf	Setosa, Versicolour, Virginica	3	

Ljubljana breast cancer veri kümesi (LBC): LBC veri kümesi 286 örnekten oluşmaktadır ve iki sınıflı bir veri kümesidir. Örneklerin % 70'i bir sınıfa, kalan % 30'u ise diğer sınıfa aittir. LBC veri kümesi 9 kategorik değişken içermektedir ve veri kümesinde 9 eksik değerli örnek yer almaktadır. İkili forma dönüştürme işleminden sonra 51 ikili girdi değişkeni elde edilmiştir. İlgili girdilere ait bilgiler Tablo 7.5.'de verilmektedir.

Tablo 7.5. LBC veri kümesi değişkenleri.

Değişken	**Değişken Bilgisi**	**İkili Değişken Sayısı**	**Alt-Aralık**
Age	10-19, 20-29, 30-39, 40-49, 50-59, 60-69, 70-79, 80-89, 90-99	9	{10-19}, {20-29}, {30-39}, {40-49}, {50-59}, {60-69}, {70-79}, {80-89},{90-99}
Menopause	lt40, ge40, premeno	3	{lt40}, {ge40}, {premeno}
Tumor-size	0-4, 5-9, 10-14, 15-19, 20-24, 25-29, 30-34, 35-39, 40-44, 45-49, 50-54, 55-59	12	{0-4}, {5-9}, {10-14}, {15-19}, {20-24}, {25-29}, {30-34}, {35-39}, {40-44}, {45-49}, {50-54}, {55-59}
Inv-nodes	0-2, 3-5, 6-8, 9-11, 12-14, 15-17, 18-20, 21-23, 24-26, 27-29, 30-32, 33-35, 36-39	13	{0-2}, {3-5}, {6-8}, {9-11}, {12-14}, {15-17}, {18-20}, {21-23}, {24-26}, {27-29}, {30-32}, {33-35}, {36-39}
Node-caps	Evet, Hayır	2	{evet}, {hayır}
Deg-malig	1, 2, 3	3	{1}, {2}, {3}
Breast	Sol, Sağ	2	{sol}, {sağ}
Breast-quad	Sol-üst, Sol-alt, Sağ-üst, Sağ-alt, Orta	5	{sol-üst}, {sol-alt}, {sağ-üst}, {sağ-alt}, {orta}
Irradiat	Evet, hayır	2	{evet}, {hayır}
Sınıf	No-recurrence-events, recurrence-events	2	

Monks-2 veri kümesi: Monks problemleri, öğrenme algoritmalarının ilk uluslararası karşılaştırılmasının temelini oluşturur ve robotların 6 farklı değişkenle tanımlandığı

yapay robot alanıyla ilgili veriler içerir. Veri kümesi 432 örnek ve 6 kategorik değişken içermektedir. Monks problemleri orjinalinde eğitim ve test kümelerine ayrılmıştır. Monks-2'de eğitim için 169, test için 432 örnek yer almaktadır. İkili forma dönüştürüldükten sonra Monks-2 veri kümesi 17 ikili girdi değişkeniyle ifade edilmektedir. Tablo 7.6. veri kümesi değişkenlerini göstermektedir.

Tablo 7.6. Monks-2 veri kümesi değişkenleri.

Değişken	**Değişken Bilgisi**	**İkili Değişken Sayısı**	**Alt-Aralık**
A1	1, 2, 3	3	{1}, {2}, {3}
A2	1, 2, 3	3	{1}, {2}, {3}
A3	1, 2	2	{1}, {2}
A4	1, 2, 3	3	{1}, {2}, {3}
A5	1, 2, 3, 4	4	{1}, {2}, {3}, {4}
A6	1, 2	2	{1}, {2}
Sınıf	0, 1	2	

Nursery veri kümesi: Nursery veri kümesi, hemşirelik okullarındaki uygulamaları derecelendirmek için geliştirilmiş hiyerarşik bir karar modelinden faydalanılarak oluşturulmuştur. 12.960 örnek ve 8 kategorik değişkenden oluşmaktadır. Veri kümesinde eksik değerli örnek bulunmamaktadır. 12.960 örneğin % 33.33'ü birinci sınıfa, % 0.015'i ikinci sınıfa, % 2.531'i üçüncü sınıfa, % 32.917'si dördüncü sınıfa ve % 31.204'ü beşinci sınıfa aittir. İkili forma dönüştürme işleminden sonra 27 ikili girdi değişkeni elde edilmiştir. Veri kümesi değişkenleri Tablo 7.7.'de verilmektedir.

Tablo 7.7. Nursery veri kümesi değişkenleri.

Değişken	**Değişken Bilgisi**	**İkili Değişken Sayısı**	**Alt-Aralık**
Parents	Usual, pretentious, great_pret	3	{usual}, {pretentious}, {great_pret}
Has_nurs	Proper, less_proper, improper, critical, very_crit	5	{proper}, {less_proper}, {improper}, {critical}, {very_crit}
form	Complete, completed, incomplete,foster	4	{complete}, {completed}, {incomplete}, {foster}
children	1, 2, 3, more	4	{1}, {2}, {3}, {more}
housing	Convenient, less_conv, critical	3	{convenient}, {less_conv}, {critical}
finance	Convenient, inconv	2	{convenient}, {inconv}
social	Non-prob, slightly_prob, problematic	3	{non-prob}, {slightly_prob}, {problematic}
health	Recommended, priority, not_recom	3	{recommended}, {priority}, {not_recom}
Sınıf	Not_recom, recommend, very_recom, priority, spec_prior	5	

Pima Indians diabetes veri kümesi: Pima veri kümesi 768 örnekten oluşmaktadır. Veri kümesi 8 sürekli değişken içermekte ve eksik değer bulundurmamaktadır. Örneklerin % 65'i birinci sınıfa ait, kalan % 35'i ise ikinci sınıfa ait veriler içermektedir. Kesiklendirme ve ikili forma dönüştürme işleminden sonra 32 ikili girdi değişkeni elde edilmiştir. İlgili girdilere ait bilgiler Tablo 7.8.'de verilmektedir.

Tablo 7.8. Pima veri kümesi değişkenleri.

Değişken	**Değişken Bilgisi**	**İkili Değişken Sayısı**	**Alt-Aralık**
Number of times pregnant	Sürekli	4	[-; 4.24), [4.25; 8.5), [8.5; 12.75), [12.65; -)
Plasma glucose concentration	Sürekli	4	[-; 49.75), [49.75; 99.5), [99.5; 149.25), [149.25; -)
Diastolic blood pressure	Sürekli	4	[-; 30.5), [30.5; 61), [61; 91.5), [91.5; -)
Triceps skin fold thickness	Sürekli	4	[-; 24.75), [24.75; 49.5), [49.5; 74.25), [74.25; -)
2-hour serum insulin	Sürekli	4	[-; 211.5), [211.5; 423), [423; 634.5), [634.5; -)
Body mass index	Sürekli	4	[-; 16.775), [16.775; 33.55), [33.55; 50.325), [50.325; -)
Diabetes pedigree function	Sürekli	4	[-; 0.6635), [0.6635; 1.249), [1.249; 1.8345), [1.8345; -)
Age	Sürekli	4	[-; 36), [36; 51), [51; 66), [66; -)
Sınıf	0, 1	2	

Spect-heart veri kümesi: Spect-heart veri kümesi, SPECT görüntülerinin teşhisini tanımlamaktadır. Her bir hasta normal veya abnormal olan iki sınıftan birine sınıflandırılmıştır. Veri kümesinde orijinal SPECT görüntülerini özetleyen 267 örnek ve 22 değişken yer almaktadır. İkili forma dönüştürme işleminden sonra 44 girdi değişkeni elde edilmiştir. Veri kümesinde eksik değer içeren örnek yer almamaktadır. Örneklerin % 21'i birinci sınıfı, kalan % 79'u ise ikinci sınıfı ifade etmektedir. Spect-heart veri kümesi orjinalinde eğitim ve test kümelerine ayrılmıştır. Eğitim kümesinde 80, test kümesinde ise 187 örnek yer almaktadır. Tablo 7.9. spect-heart veri kümesi değişkenlerini tanımlamaktadır.

Tablo 7.9. Spect-heart veri kümesi değişkenleri.

Değişken	Değişken Bilgisi	İkili Değişken Sayısı	Alt-Aralık
F1	1, 0	2	{1}, {0}
F2	1, 0	2	{1}, {0}
F3	1, 0	2	{1}, {0}
F4	1, 0	2	{1}, {0}
F5	1, 0	2	{1}, {0}
F6	1, 0	2	{1}, {0}
F7	1, 0	2	{1}, {0}
F8	1, 0	2	{1}, {0}
F9	1, 0	2	{1}, {0}
F10	1, 0	2	{1}, {0}
F11	1, 0	2	{1}, {0}
F12	1, 0	2	{1}, {0}
F13	1, 0	2	{1}, {0}
F14	1, 0	2	{1}, {0}
F15	1, 0	2	{1}, {0}
F16	1, 0	2	{1}, {0}
F17	1, 0	2	{1}, {0}
F18	1, 0	2	{1}, {0}
F19	1, 0	2	{1}, {0}
F20	1, 0	2	{1}, {0}
F21	1, 0	2	{1}, {0}
F22	1, 0	2	{1}, {0}
Overall Diagnosis (Sınıf)	0, 1	2	

Tic-Tac-Toe endgame veri kümesi: Tic-Tac-Toe veri kümesi, tic-tac-toe oyunundaki olası konfigürasyonların tamamını içermektedir. Çıktı değişkeni, oyunu kazanmayla ilgili sonucu vermektedir. Veri kümesinde 9 kategorik girdi ve 958 örnek vardır. Eksik değerli değişken bulunmamaktadır. Örneklerin % 65.3'ü kazanma sınıfını, kalanı ise kaybetme sınıfını ifade etmektedir. İkili forma dönüştürüldüğünde 27 ikili girdi elde edilmiştir. Tablo 7.10. tic-tac-toe veri kümesi değişkenlerini açıklamaktadır.

Tablo 7.10. Tic-Tac-Toe veri kümesi değişkenleri.

Değişken	**Değişken Bilgisi**	**İkili Değişken Sayısı**	**Alt-Aralık**
Top-left-square	x, o, b	3	{x}, {o}, {b}
Top-middle-square	x, o, b	3	{x}, {o}, {b}
Top-right-square	x, o, b	3	{x}, {o}, {b}
Middle-left-square	x, o, b	3	{x}, {o}, {b}
Middle-middle-square	x, o, b	3	{x}, {o}, {b}
Middle-right-square	x, o, b	3	{x}, {o}, {b}
Bottom-left-square	x, o, b	3	{x}, {o}, {b}
Bottom-middle-square	x, o, b	3	{x}, {o}, {b}
Bottom-right-square	x, o, b	3	{x}, {o}, {b}
Sınıf	Pozitif, negatif	2	

Vote veri kümesi: Vote veri kümesi, 16 anahtar oy üzerindeki, oylamalar ile ilgili verileri içermektedir. Veri kümesinde 435 örnek ve 16 kategorik girdi yer almaktadır. Örneklerin % 45.2'si birinci sınıfı, kalan %54.8'i ise ikinci sınıfı tanımlamaktadır. Vote veri kümesinin çeşitli örneklerinde eksik değerli değişkenler yer almaktadır. İkili forma dönüştürme işleminden sonra 32 ikili girdi değişkeni elde edilmiştir. Tablo 7.11. vote veri kümesi değişkenlerini açıklamaktadır.

Tablo 7.11. Vote veri kümesi değişkenleri.

Değişken	Değişken Bilgisi	İkili Değişken Sayısı	Alt-Aralık
Handicapped-infants	Evet, hayır	2	{evet}, {hayır}
Water-project-cost-sharing	Evet, hayır	2	{evet}, {hayır}
Adoption-of-the-budget-resolution	Evet, hayır	2	{evet}, {hayır}
Physician-fee-freeze	Evet, hayır	2	{evet}, {hayır}
El-salvador-ait	Evet, hayır	2	{evet}, {hayır}
Religious-groups-in-schools	Evet, hayır	2	{evet}, {hayır}
Anti-satellite-test-ban	Evet, hayır	2	{evet}, {hayır}
Aid-to-nicaraguan-contras	Evet, hayır	2	{evet}, {hayır}
Mx-missile	Evet, hayır	2	{evet}, {hayır}
Immigration	Evet, hayır	2	{evet}, {hayır}
Synfuels-corporation-cutback	Evet, hayır	2	{evet}, {hayır}
Education-spending	Evet, hayır	2	{evet}, {hayır}
Superfund-right-to-sue	Evet, hayır	2	{evet}, {hayır}
Crime	Evet, hayır	2	{evet}, {hayır}
Duty-free-exports	Evet, hayır	2	{evet}, {hayır}
Export-administration-act-south-africa	Evet, hayır	2	{evet}, {hayır}
Sınıf	Democrat, republican	2	

Wisconsin breast cancer veri kümesi (WBC): Bu veri kümesi 699 örnekten oluşmakta ve her bir örnek 9 sürekli değişken içermektedir. WBC veri kümesinde, 16 eksik değerli örnek mevcuttur. Örneklerin % 65.5'i birinci grubu, % 34.5'i ise ikinci grubu tanımlamaktadır. Kesiklendirme ve ikili forma dönüştürme işleminden sonra 27 ikili girdi değişkeni elde edilmiştir. İlgili girdilere ait bilgiler Tablo 7.12.'de gösterilmektedir.

Tablo 7.12. WBC veri kümesi değişkenleri.

Değişken	Değişken Bilgisi	İkili Değişken Sayısı	Alt-Aralık
Clump thickness	Tamsayı (1-10)	3	[-; 4), [4; 7), [7; -)
Uniformity of cell size	Tamsayı (1-10)	3	[-; 4), [4; 7), [7; -)
Uniformity of cell shape	Tamsayı (1-10)	3	[-; 4), [4; 7), [7; -)
Marginal adhesion	Tamsayı (1-10)	3	[-; 4), [4; 7), [7; -)
Single epithelial cell size	Tamsayı (1-10)	3	[-; 4), [4; 7), [7; -)
Bare nuclei	Tamsayı (1-10)	3	[-; 4), [4; 7), [7; -)
Bland chromatin	Tamsayı (1-10)	3	[-; 4), [4; 7), [7; -)
Normal nucleoli	Tamsayı (1-10)	3	[-; 4), [4; 7), [7; -)
Mitoses	Tamsayı (1-10)	3	[-; 4), [4; 7), [7; -)
Sınıf	Benign (2), Malignant (4)	2	

Zoo veri kümesi: Zoo veri kümesi, 17 kategorik değişken içermektedir. Ancak bu değişkenlerden "animal name" her örnek için farklı bir değer içerdiğinden dolayı [179]'da olduğu gibi veri kümesinden çıkarılmıştır. 7 sınıftan oluşan zoo veri kümesinde 101 örnek yer almaktadır. Örnek sayısının az ve sınıf sayısın çok olmasından dolayı örneklerin % 67'si eğitim, kalan % 33'ü ise test kümesi olarak kullanılmıştır. Örneklerin %41'i birinci sınıfa, % 20'si ikinci sınıfa, % 5'i üçüncü sınıfa, % 13'ü dördüncü sınıfa, % 4'ü beşinci sınıfa, % 8'i altıncı sınıfa ve % 10'u yedinci sınıfa aittir. Veri kümesi herhangi bir boş değer içermemektedir. İkili forma dönüştürme işleminden sonra veri kümesinde 36 girdi değişkeni yer almaktadır. Tablo 7.13. ilgili değişkenleri açıklamaktadır.

Tablo 7.13. Zoo veri kümesi değişkenleri

Değişken	Değişken Bilgisi	İkili Değişken Sayısı	Alt-Aralık
Hair	0, 1	2	{0}, {1}
Feathers	0, 1	2	{0}, {1}
Eggs	0, 1	2	{0}, {1}
Milk	0, 1	2	{0}, {1}
Airbone	0, 1	2	{0}, {1}
Aquatic	0, 1	2	{0}, {1}
Predator	0, 1	2	{0}, {1}
Toothed	0, 1	2	{0}, {1}
Backbone	0, 1	2	{0}, {1}
Breathes	0, 1	2	{0}, {1}
Venomous	0, 1	2	{0}, {1}
Fins	0, 1	2	{0}, {1}
Legs	0, 2, 4, 5, 6, 8	2	{0}, {2}, {4}, {5}, {6}, {8}
Tail	0, 1	2	{0}, {1}
Domestic	0, 1	2	{0}, {1}
Catsize	0, 1	2	{0}, {1}
Type (Sınıf)	1, 2, 3, 4, 5, 6, 7	2	

7.3. Performans Ölçütleri

Test doğruluğu ve ortalama kural sayısı geliştirilen algoritmanın performansını değerlendirmek için kullanılmıştır. Doğruluk, sınıflandırıcının doğru sonuçlar üretmekteki yeteneğini ölçen bir ölçüttür ve (7.1.) eşitliği ile hesaplanır.

$$Doğruluk = \frac{Test\ veri\ kümesinde\ sağlanan\ örnek\ sayısı}{Test\ kümesindeki\ toplam\ örnek\ sayısı} \quad (7.1.)$$

Monks-2, Spect-heart ve Zoo veri kümelerinin tek bir eğitim ve test kümesine ayrılmalarından dolayı, bu veri kümelerinde doğruluk aynı eğitim ve test kümelerinde algoritmanın 10 kez çalıştırılması ile elde edilmiştir. Bütün veri kümeleri için tahminleyici doğruluklar, 10 çalıştırmadaki test doğruluklarının ortalamasının alınması ile belirlenmiştir. Ayrıca tahminleyici doğrulukların standart sapmaları da hesaplanmıştır.

Kural sayısı ise kısaca üretilen kural kümesindeki kural sayısı ile ifade edilmektedir. Bütün veri kümelerinde, 10 katlı çapraz doğrulama sonucu elde edilen kural kümelerindeki kural sayılarının ortalamaları alınarak ortalama kural sayıları hesaplanmıştır.

7.4. Deney Tasarımı

Geliştirilen eğitilmiş yapay sinir ağlarından karınca koloni optimizasyonu ile sınıflandırma kuralları çıkarımı algoritmasının etkili faktör kümelerini belirlemek için L-36 ($2^{**9}3^{*1}$) Taguchi deney tasarımı [180] gerçekleştirilmiştir. Tasarımda Minitab 14 istatistiksel yazılımı kullanılmıştır. Örnek uzayın bir alt kümesi üzerinde analiz gerçekleştiren Taguchi tasarımı kullanılmasının nedeni, tasarımda yer alacak faktör sayısının, dolayısıyla programın çalıştırılma sayısının ($2^9*3^1=1536$) çok fazla olmasıdır. Tahminleyici doğruluğun ölçülmesinde 10-katlı çapraz geçerlilik testinin uygulanması ve 12 veri kümesinin analiz kapsamına alınması bu sayıyı 120 kat (1536*10*12=184.320) daha artırmaktadır. L-36 Taguchi tasarımı ile her bir veri kümesi için geliştirilen algoritma 36*10=360 kez çalıştırılmıştır. Dolayısıyla örnek uzayından 360'lık bir alt-küme alınarak analizler gerçekleştirilmiştir.

Deney tasarımında etkinlik ölçütü olarak "*test doğruluğu (T.D.)*" ele alınmıştır. Test doğruluğu (7.1.) eşitliği ile hesaplanmaktadır. Taguchi yöntemi ile faktörler, etkinlik ölçütü üzerindeki etkilerine göre sıralanmıştır.

Taguchi tasarımında test doğruluğu üzerinde faktörlerin etkileri analiz edilirken, ana etkiler ve ikili etkileşimler dikkate alınmıştır. Çalışılacak çok sayıda tasarım faktörü varsa çok kullanışlı olan ana etki grafikleri, düzey ortalamalarını karşılaştırmaya ve etkinlik ölçütünü en çok etkileyen faktörün ayırt edilmesine yardımcı olmaktadır [181].

Ana etki grafiğinde elde edilen doğru x-eksenine paralel ise, faktörün ele alınan her düzeyi ölçütü aynı yönde etkiliyor demektir ve bu da ana etki yoktur şeklinde yorumlanır. Eğer elde edilen doğru belirli bir eğime sahipse, bu durumda bir ana etki mevcuttur ve ele alınan faktörün farklı düzeyleri, etkinlik ölçütünü farklı şekilde etkilemektedir. Doğrudaki eğim arttıkça etkinin önemi de artar. Ana etki grafiklerinde, doğruların eğimleri karşılaştırılarak, faktörün göreceli önemi belirlenebilir.

Ancak ana etki grafikleri faktörlerin etkinlikleri hakkında net bir sonuca varmak için yeterli değildir. Çünkü etkileşim eğrileri, ana etkiyi doğrulayabilecekleri gibi ortadan da kaldırabilirler. Bu bakımdan etkileşim grafiklerinin de incelenmesi gerekir.

Bir etkileşim grafiği bir faktörün ayarlamalarını diğer bir faktöre değiştirmekteki etkiyi gösterir. Etkileşim grafikleri ana etkiyi doğrulayabileceğinden veya ortadan kaldırabileceğinden dolayı, etkileşimleri değerlendirmek çok önemlidir. Eğer etkileşim grafiklerinde ele alınan faktörlerin doğruları paralele yakınsa, bu durumda bu faktörler birbirinden bağımsızdır ve çok az veya hiç etkileşim yoktur şeklinde yorumlanır [182]. Kesişen doğrular, faktörler arasındaki güçlü etkileşimleri ifade ederler.

Taguchi tasarımı sonucu ana etkiler ve ikili etkileşim grafiklerinden faydalanılarak test doğruluğu üzerinde etkisi olan faktörler ve bu faktörlerin etkin düzeyleri belirlenmiştir. Geliştirilen kural çıkarım algoritması, her bir veri kümesi için faktörlerin belirlenen düzeylerinde çalıştırılmış ve etkinlik ölçütleri hesaplanmıştır.

7.4.1. Uygun Faktör Kümesinin Belirlenmesi

Tasarımda ilk olarak uygun faktör kümesi ve bu faktörlerin tasarımda kullanılacak düzeyleri belirlenmiştir. TAKKOYSA-sınıflandırıcısında, karınca sayısı, TAKKO iterasyon sayısı, sıklık faktörü, feremon sabiti, buharlaşma katsayısı, momentum katsayısı, öğrenme katsayısı, ÇKA iterasyon sayısı, gizli katman sayısı ve gizli katman(lar)daki işlem elemanı sayısı olmak üzere 10 farklı kontrol faktörü vardır.

Geliştirilen algoritma farklı test problemleri üzerinde pek çok kez çalıştırılarak tasarımda kullanılacak uygun faktör düzeyleri ve düzey sayıları belirlenmiştir. Bölüm 3'de de değinildiği gibi YSA'nda en uygun gizli katman sayısının ve gizli katman(lar)daki işlem elemanı sayılarının belirlenmesine yönelik kesin bir yöntem yoktur. Ancak bir veya iki gizli katman yaygın olarak kullanılmaktadır, bu nedenle gizli katman sayısı olarak tasarımda bir veya iki gizli katman dikkate alınmıştır. Gizli katman(lar)daki işlem elemanı sayısı faktörünün düzeyleri ise (7.2)-(7.4) eşitlikleri kullanılarak belirlenmiştir [179].

$$h = 2rand\sqrt{v \times c} \qquad (7.2.)$$

$$h_1 = \frac{\sqrt{v \times c}}{2} + 4rand\frac{\sqrt{v \times c}}{c} \quad (7.3.)$$

$$h_2 = rand\frac{\sqrt{v \times c}}{c} + c \quad (7.4.)$$

(7.2.)-(7.4.) eşitliklerinde *v* girdi değişken sayısını, *c* sınıf sayısını göstermektedir. *Rand* [0,1] aralığında değer alan rastgele bir sayıdır. Tek gizli katman içeren yapay sinir ağları için (7.2.) eşitliği kullanılır. İki gizli katmanlı YSA'lar için, birinci gizli katmandaki işlem elemanı sayıları eşitlik (7.3.) kullanılarak, ikinci gizli katmandaki işlem eleman sayıları ise (7.4.) eşitliği kullanılarak belirlenir.

Tasarımda kullanılan faktör ve faktör düzeyleri Tablo 7.14.'de verilmektedir.

Tablo 7.14. Deney tasarımı faktörleri ve düzeyleri.

Faktörler	Açıklama	Düzey Sayısı	Düzeyler
M	Karınca sayısı	2	100 200
T	TAKKO iterasyon sayısı	2	500 1000
f	Sıklık faktörü	2	2 5
Q	Feremon sabiti	2	5 10
ρ	Buharlaşma katsayısı	2	0,8 0,9
α	Momentum katsayısı	2	0,7 0,9
λ	Öğrenme katsayısı	2	0,2 0,4
Epoch	ÇKA iterasyon sayısı	2	10000 20000
GK	Gizli katman sayısı	2	1 2
İES	GK(lar)daki işlem elemanı sayısı	3	(7.3)-(7.5) eşitlikleri

7.4.2. Analiz Sonuçları

Geliştirilen algoritmada analizler her veri kümesi için ayrı yapılmıştır. Deneysel tasarımda analizlerin ayrı ayrı ele alınmasındaki ana neden, gizli katman(lar)daki işlem elemanı sayılarının problem parametrelerine bağlı olarak (7.2)-(7.4) eşitlikleri ile belirlenmesi, dolayısıyla her veri kümesinde farklı işlem elemanı sayılarının ele alınmasıdır. Diğer bir neden ise veri kümesindeki örnek sayısının değişmesi ile faktörlerin etkinliğinin değişebileceğidir.

Tablo 7.15. hazırlanan örnek deney tasarımıyla ilgili faktörler ve bu faktörlerin düzeylerinin listesini göstermektedir. İES sütunundaki 1, 2 ve 3 ifadeleri (7.2.)-(7.4.) eşitlikleri kullanılarak belirlenen gizli katman(lar)daki işlem elemanı sayılarını ifade etmektedir. Tabloda bu şekilde verilmesinin nedeni, bu faktörlerin düzeylerinin her veri kümesinde farklı değerler almasıdır. Ayrıca gizli katman sayısının 1 veya 2 olmasına göre kullanılan eşitlik ve hesaplanan değerler de değişmektedir.

Tablo'daki T.D. sütunu TAKKOYSA-sınıflandırıcısının ilgili test probleminde, tasarımın ilgili faktör ve faktör düzeyleriyle elde edilen test doğruluklarını ifade etmektedir. Test doğruluğu, deney tasarımında yanıt faktörü olarak ele alınmıştır.

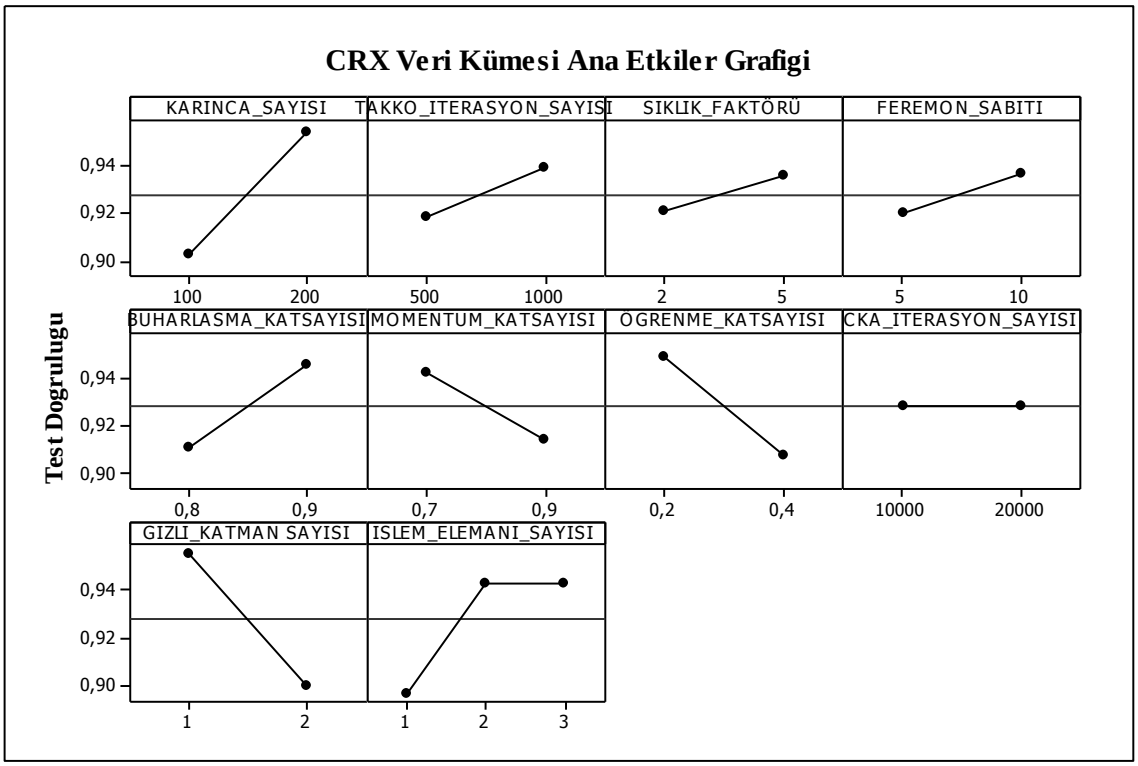
Tablo 7.15. L-36 örnek tablosu.

	M	T	f	Q	ρ	α	λ	Epoch	GK	İES	T.D.
1	100	500	2	5	0,8	0,7	0,2	10000	1	1	0,990742
2	100	500	2	5	0,8	0,7	0,2	10000	1	2	0,987732
3	100	500	2	5	0,8	0,7	0,2	10000	1	3	0,990742
4	100	500	2	5	0,8	0,9	0,4	20000	2	1	0,818289
5	100	500	2	5	0,8	0,9	0,4	20000	2	2	0,834724
6	100	500	2	5	0,8	0,9	0,4	20000	2	3	0,917131
7	100	500	5	10	0,9	0,7	0,2	10000	2	1	0,990741
8	100	500	5	10	0,9	0,7	0,2	10000	2	2	0,986114
9	100	500	5	10	0,9	0,7	0,2	10000	2	3	0,961576
10	100	1000	2	10	0,9	0,7	0,4	20000	1	1	0,986576
11	100	1000	2	10	0,9	0,7	0,4	20000	1	2	0,984725
12	100	1000	2	10	0,9	0,7	0,4	20000	1	3	0,985649
13	100	1000	5	5	0,9	0,9	0,2	20000	1	1	0,986808
14	100	1000	5	5	0,9	0,9	0,2	20000	1	2	0,986345
15	100	1000	5	5	0,9	0,9	0,2	20000	1	3	0,987965
16	100	1000	5	10	0,8	0,9	0,4	10000	2	1	0,797919
17	100	1000	5	10	0,8	0,9	0,4	10000	2	2	0,866904
18	100	1000	5	10	0,8	0,9	0,4	10000	2	3	0,860886
19	200	500	5	10	0,8	0,7	0,4	20000	1	1	0,893751
20	200	500	5	10	0,8	0,7	0,4	20000	1	2	0,987966
21	200	500	5	10	0,8	0,7	0,4	20000	1	3	0,984029
22	200	500	5	5	0,9	0,9	0,4	10000	1	1	0,969908
23	200	500	5	5	0,9	0,9	0,4	10000	1	2	0,987037
24	200	500	5	5	0,9	0,9	0,4	10000	1	3	0,990512
25	200	500	2	10	0,9	0,9	0,2	20000	2	1	0,718983
26	200	500	2	10	0,9	0,9	0,2	20000	2	2	0,988891
27	200	500	2	10	0,9	0,9	0,2	20000	2	3	0,844877
28	200	1000	5	5	0,8	0,7	0,2	20000	2	1	0,960187
29	200	1000	5	5	0,8	0,7	0,2	20000	2	2	0,956484
30	200	1000	5	5	0,8	0,7	0,2	20000	2	3	0,886577
31	200	1000	2	10	0,8	0,9	0,2	10000	1	1	0,984261
32	200	1000	2	10	0,8	0,9	0,2	10000	1	2	0,982871
33	200	1000	2	10	0,8	0,9	0,2	10000	1	3	0,991436
34	200	1000	2	5	0,9	0,7	0,4	10000	2	1	0,926391
35	200	1000	2	5	0,9	0,7	0,4	10000	2	2	0,980094
36	200	1000	2	5	0,9	0,7	0,4	10000	2	3	0,951389

7.4.2.1. CRX Veri Kümesi Deney Tasarımı

Deney tasarımının ilk aşamasında Tablo 7.14'de verilen faktör ve faktör düzeyleri ile Taguchi tasarımı gerçekleştirilmiştir. Taguchi deney tasarımı sonucunda etkinlik ölçütü üzerinde etkili faktörler belirlenmiş ve sıralanmıştır.

CRX veri kümesi için hazırlanan deney tasarımının analizi ile elde edilen ana etkiler grafiği Şekil 7.1.'de gösterilmiştir.



Şekil 7.1. CRX veri kümesi ana etkiler grafiği.

Taguchi ana etkiler grafiği, faktörlerin etkinlik ölçütü üzerindeki etki düzeylerini göstermektedir. Grafik üzerinde eğimi yüksek olan doğrulara ait faktörlerin, etkinlik ölçütü üzerindeki etkileri fazladır. Bu etki doğrunun faktörlerin düzeyleri arasındaki durumuna göre etkinlik ölçütü üzerinde pozitif veya negatif etki olarak yorumlanabilir. X-eksenine paralel olan doğrularda ise etki yoktur.

Şekil 7.1.'de verilen ana etki grafikleri incelendiğinde gizli katman sayısı faktörünün test doğruluğu etkinlik ölçütü üzerinde etkili olduğu görülmektedir. Bu etkinin yönü GK

faktörünün birinci ve ikinci düzeyleri arasında test doğruluğunu azaltan yöndedir. Yani bir gizli katman kullanıldığında test doğruluğu yüksek iken; gizli katman sayısının ikiye çıkarılması test doğruluğunu negatif yönde etkilemekte ve düşürmektedir. Dolayısıyla bu veri kümesinde, Taguchi deney tasarımına göre gizli katman sayısı ilk düzeyinde iken en iyi sonucu vermiştir. Benzer şekilde momentum katsayısı ve öğrenme katsayısı faktörlerinde de etki negatif yöndedir ve bu faktörlerin de ilk düzeyleri en iyi sonucu vermektedir.

Karınca sayısı, TAKKO iterasyon sayısı, sıklık faktörü, feremon sabiti ve buharlaşma katsayısı faktörlerinin ana etki eğrileri dikkate alınacak olursa bu faktörlerin etkinlik ölçütü üzerinde etkin oldukları söylenebilir. Çünkü doğrular belirli eğimlere sahiptir. Ayrıca bu etki test doğruluğunu artıracak şekilde pozitif yöndedir. Bu faktörlerin düzeylerindeki artış etkinlik ölçütünde de artışa neden olacaktır. Dolayısıyla bu faktörlerin ikinci düzeylerinin test doğruluğu üzerinde en iyi sonucu verdiği söylenebilir.

İşlem elemanı sayısı faktörünün üç düzeyi bulunmaktadır. CRX veri kümesi için bu faktörün birinci düzeyinden ikinci düzeyine geçiş test doğruluğu etkinlik ölçütü üzerinde pozitif yönde bir etki yaratmıştır. Üçüncü düzeyin ise etkinlik yönünden ikinci düzeyden farkı yoktur, çünkü ikinci düzey ile üçüncü düzey arasındaki doğru x-eksenine paralel yöndedir. Bu faktörün ikinci veya üçüncü düzeyinin kullanımı test doğruluğunu artırmaktadır.

Şekil 7.1.'de ÇKA iterasyon sayısı faktörünün eğrisi incelenecek olursa, bu eğrinin x-eksenine paralel olduğu yani herhangi bir ana etki göstermediği söylenebilir. Dolayısıyla test doğruluğu etkinlik ölçütü üzerinde ÇKA iterasyon sayısı faktörünün hiçbir etkisi yoktur, hangi düzeyi kullanılırsa kullanılsın, sonuç değişmeyecektir.

CRX veri kümesi için Taguchi deney tasarımı sonucunda elde edilen, faktörlerin etkinlik ölçütü üzerindeki etkilerine göre sıralamaları, ortalamalar ile birlikte Tablo 7.16.'da verilmektedir. Tabloda gösterilen fark, etki derecesini ölçen bir değerdir ve bu değer ne kadar yüksek ise ait olduğu faktörün etkisi o derece yüksektir. Sıralama satırı ise faktörlerin etkilerine göre sıralama derecelerini göstermektedir.

Tablo 7.16. CRX veri kümesi için Taguchi faktör sıralaması.

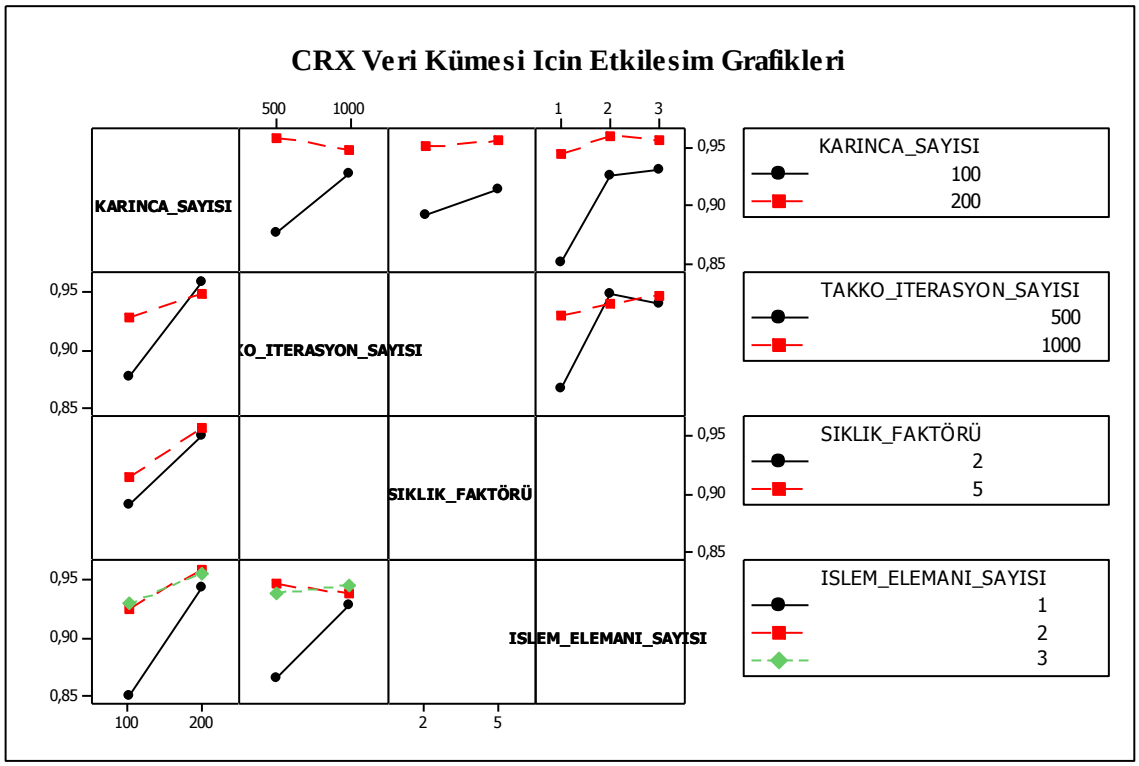
Faktörler / Düzeyleri	M	T	f	Q	ρ	α	λ	Epoch	GK	İES
1	0,9024	0,9176	0,9208	0,9194	0,9108	0,9419	0,9483	0,9283	0,9553	0,8975
2	0,9535	0,9383	0,9351	0,9366	0,9451	0,9140	0,9076	0,9276	0,9007	0,9433
3										0,9431
Fark	0,0511	0,0207	0,0143	0,0172	0,0342	0,0280	0,0408	0,0007	0,0546	0,0459
Sıralama	2	7	9	8	5	6	4	10	1	3

CRX veri kümesi için Tablo 7.16. ve Şekil 7.1.'den elde edilen bilgilere göre, bu veri kümesinde faktörlerin test doğruluğu üzerindeki etkinliklerine göre sıralaması ve en iyi sonuç veren düzeyleri Tablo 7.17.'de gösterilmektedir. ÇKA iterasyon sayısı faktörü, test doğruluğu üzerinde etkiye sahip olmadığı için Tablo'da yer almamaktadır.

Tablo 7.17. CRX veri kümesi için faktör sıralamaları ve düzeyleri

Faktör	GK	M	İES	λ	ρ	α	T	Q	f
Düzey	1	200	12/19	0,2	0,9	0,7	1000	10	5

Daha önce de değinildiği gibi ana etki grafikleri faktörlerin etkileri hakkında net bir sonuca varmak için yeterli değildir, etkileşimlerin de incelenmesi gerekir. Çünkü etkileşim eğrileri, ana etkiyi doğrulayabilecekleri gibi ortadan da kaldırabilirler. Şekil 7.2. CRX problemi için ikili etkileşim grafiklerini göstermektedir.



Şekil 7.2. CRX veri kümesi etkileşim grafiği.

Şekil 7.2. incelenecek olursa karınca sayısı-TAKKO iterasyon sayısı ve TAKKO iterasyon sayısı-gizli katman(lar)daki işlem elemanı sayısı arasında bir etkileşim olduğu görülür. Çünkü bu faktörlerle ilgili etkileşim eğrileri birbirini kesmektedir. Kalan diğer faktörler arasında ise ya hiç etkileşim yoktur ya da doğrular kesişmediğinden dolayı etkiler yok denilecek kadar azdır. Etkileşime sahip faktörlerde ana etkiler, karar vermek için yeterli değildir. Çünkü etkileşimler ana etkiyi daha çok kuvvetlendirebileceği gibi ana etkiyi ortadan kaldırabilirler de.

Sonuç olarak, ana etkiler ile birlikte ikili etkileşimler dikkate alındığında; ana etkilere göre ikinci düzeyinde daha etkin olan TAKKO iterasyon sayısı faktörü "T", ilk düzeyinde çalıştırırsa test doğruluğu artacaktır. İkili etkileşimler bu faktörün ana etkiler ile belirlenen düzeyinde değişikliğe neden olmuştur.

Benzer şekilde ana etki grafiğine göre gizli katman(lar)daki işlem elemanı sayısı "İES" faktörü ikinci veya üçüncü düzeyinde çalıştırıldığında test doğruluğunda daha iyi çözüm elde edilmekteydi. İkili etkileşim grafiğinde bu faktörün TAKKO iterasyon sayısı faktörü ile arasındaki etkileşim eğrisi ikinci düzeyinin test doğruluğu üzerinde artışa

neden olacağını göstermektedir. Dolayısıyla İES faktörü üçüncü düzeyinde çalıştırılmalıdır. Çok katmanlı algılayıcıdaki iterasyon sayısını gösteren "Epoch" faktörünün ise hangi düzeyde çalıştırıldığının test doğruluğu üzerinde etkisi yoktur. Ana etkilerde paralel bir doğruya sahiptir ve diğer faktörler ile arasında da etkileşim çıkmamıştır. Dolayısıyla geliştirilen algoritmada bu faktör zamandan tasarruf için ilk düzeyinde çalıştırılmıştır.

Geliştirilen algoritmada CRX veri kümesi için belirlenen faktör düzeyleri Tablo 7.18.'de verilmektedir.

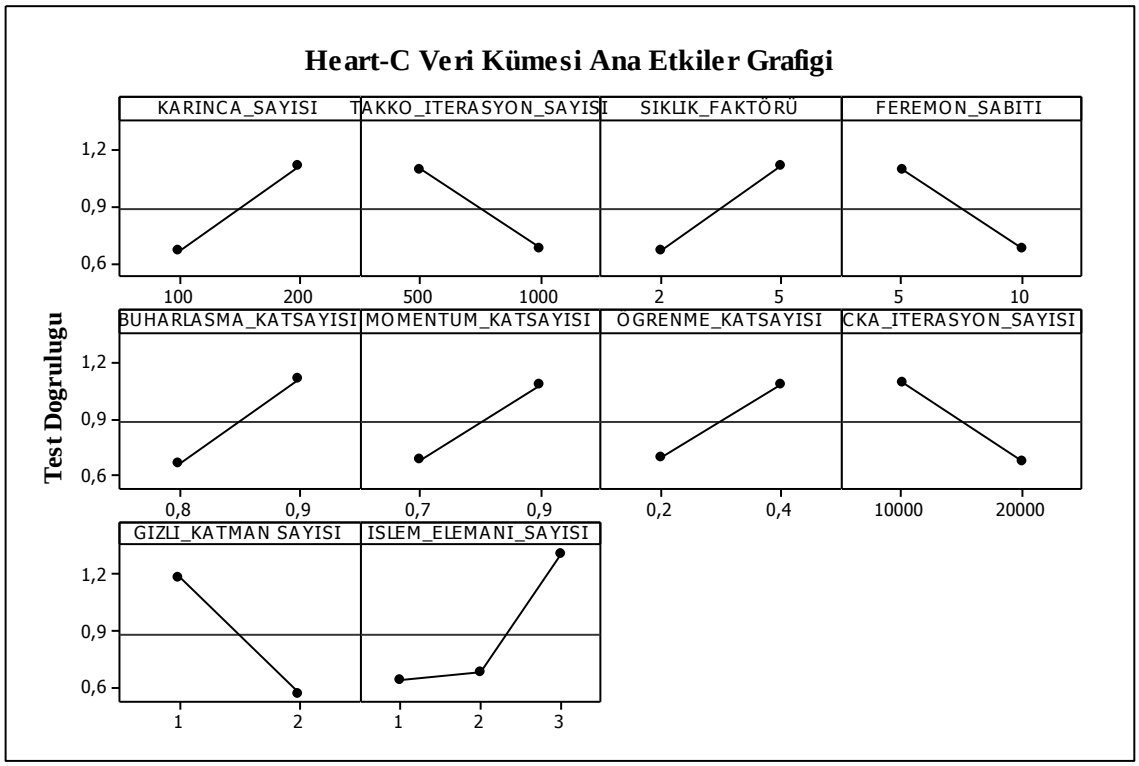
Tablo 7.18. CRX veri kümesi için belirlenen faktör düzeyleri.

Faktör	M	T	f	Q	ρ	α	λ	Epoch	GK	İES
Değer	200	500	5	10	0,9	0,7	0,2	10000	1	12

7.4.2.2. Heart-C Veri Kümesi Deney Tasarımı

Heart-C veri kümesi için hazırlanan Taguchi deney tasarımının analizi ile elde edilen ana etkiler grafiği Şekil 7.3.'de gösterilmektedir.

Şekildeki ana etki grafiklerine göre, bütün faktörlerin test doğruluğu üzerinde etkisi vardır. Karınca sayısı, sıklık faktörü, buharlaşma katsayısı, momentum katsayısı, öğrenme katsayısı ve işlem elemanı sayısı faktörlerinin etkinlik ölçütü üzerinde pozitif etkisi olduğu yani bu faktörlerin düzeylerinin ilk değerinden ikinci değerine çıkarılmasının test doğruluğunu artıracağı söylenebilir. Bu faktörler son düzeylerinde çalıştırılmalıdır. Kalan diğer faktörler için ise tam tersi bir durum söz konusudur. Faktör değerlerinin ikinci düzeylerine çıkarılması test doğruluğunu düşürecektir; dolayısıyla kalan faktörler ilk düzeylerinde çalıştırılmalıdır. Böylece etkinlik ölçütü olan test doğruluğu üzerinde en iyi sonuç elde edilebilir.



Şekil 7.3. Heart-C veri kümesi ana etkiler grafiği.

Heart-C veri kümesi için Taguchi deney tasarımı sonucunda elde edilen, faktörlerin etkinlik ölçütü üzerindeki etkilerine göre sıralamaları, ortalamalar ile birlikte Tablo 7.19.'da verilmektedir.

Tablo 7.19. Heart-C veri kümesi için Taguchi faktör sıralaması.

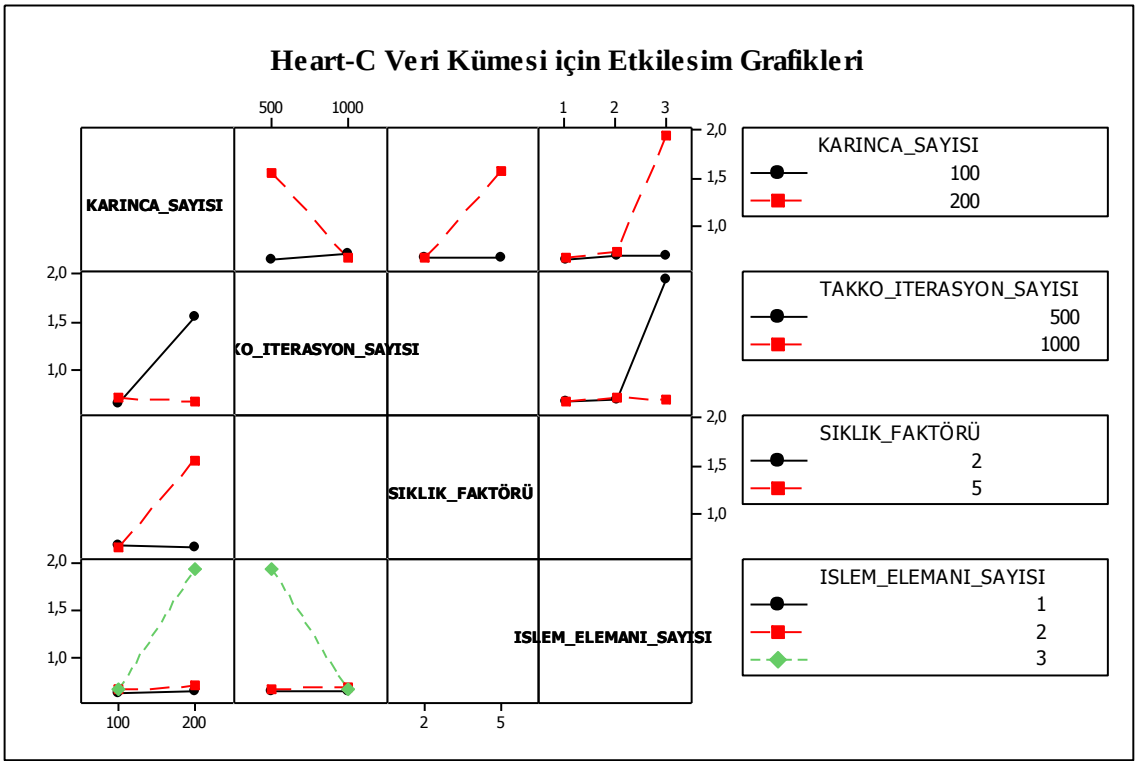
Faktörler	M	T	f	Q	ρ	α	λ	Epoch	GK	İES
Düzeyleri										
1	0,6565	1,0910	0,6551	1,0911	0,6575	0,6844	0,6872	1,0950	1,1866	0,6496
2	1,1066	0,6730	1,1090	0,6729	1,1065	1,0797	1,0769	0,6690	0,5775	0,6877
3										1,3118
Fark	0,4491	0,4180	0,4539	0,4182	0,4490	0,3953	0,3897	0,4260	0,6091	0,6652
Sıralama	4	8	3	7	5	9	10	6	2	1

Tablo 7.19. ve Şekil 7.3.'den elde edilen bilgilere göre, bu veri kümesinde faktörlerin test doğruluğu üzerindeki etkilerine göre sıralaması ve en iyi sonuç veren düzeyleri Tablo 7.20.'de gösterilmektedir.

Tablo 7.20. Heart-C veri kümesi için faktör sıralamaları ve düzeyleri.

Faktör	İES	GK	f	M	ρ	Epoch	Q	T	α	λ
Düzey	25	1	5	200	0,9	10000	5	500	0,9	0,4

Şekil 7.4. ikili etkileşim grafiklerini göstermektedir. Şekilden görülebileceği gibi faktörler arasında etkileşim söz konusudur ancak bunlar büyük etkileşimler değildir. Karınca sayısı "M" faktörünün TAKKO iterasyon sayısı "T", sıklık faktörü "f" ve işlem elemanı sayısı "İES" arasında; TAKKO iterasyon sayısı faktörünün işlem elemanı sayısı faktörü arasında ikili etkileşim söz konusudur. Kalan diğer faktörler arasında ise ya hiç etkileşim yoktur ya da doğrular kesişmediğinden dolayı etkiler yok denilecek kadar azdır. Etkileşime sahip faktörlerde ana etkiler, karar vermek için yeterli değildir.



Şekil 7.4. Heart-C veri kümesi etkileşim grafiği.

Sonuç olarak, ana etkiler ile birlikte ikili etkileşimler dikkate alındığında; etkileşimlerin ana etkileri kuvvetlendirdiği söylenebilir. Çünkü bu etkileşimler faktör düzeylerinde herhangi bir değişikliğe neden olmamakta, düzeyleri desteklemektedir.

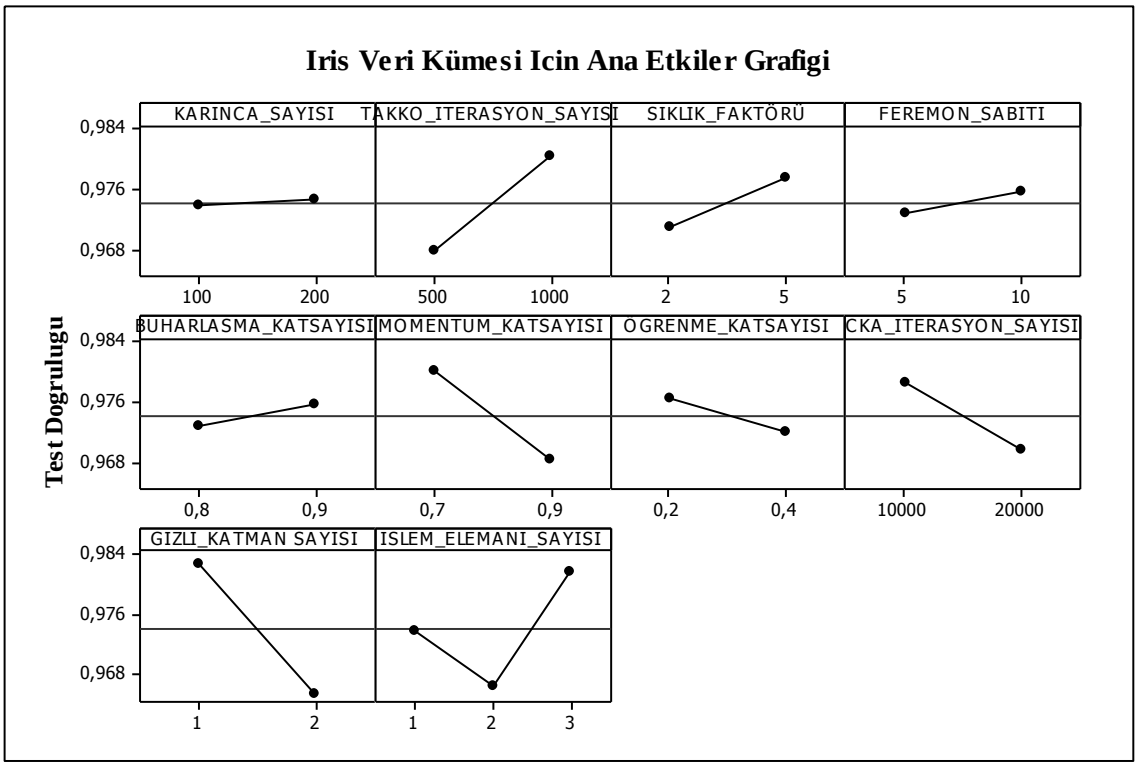
Geliştirilen algoritmada Heart-C veri kümesi için kullanılan faktör düzeyleri Tablo 7.21.'de verilmektedir.

Tablo 7.21. Heart-C veri kümesi için belirlenen faktör düzeyleri.

Faktör	M	T	f	Q	ρ	α	λ	Epoch	GK	İES
Değer	200	500	5	5	0,9	0,9	0,4	10000	1	25

7.4.2.3. Iris Veri Kümesi Deney Tasarımı

Şekil 7.5. Iris veri kümesi için Taguchi deney tasarımı ile elde edilen ana etkiler grafiğini göstermektedir.



Şekil 7.5. Iris veri kümesi ana etkiler grafiği.

Şekilden de görülebileceği gibi karınca sayısı "M" faktörünün ana etki doğrusu x-eksenine paralel olduğundan dolayı etkinlik ölçütü test doğruluğu üzerinde çok fazla etkisi yoktur. Diğer faktörler ise test doğruluğunu pozitif veya negatif yönde etkilemektedir. TAKKO iterasyon sayısı, sıklık faktörü, feremon sabiti ve buharlaşma katsayısı faktörlerinin düzeylerindeki artış test doğruluğunda artış sağlayacaktır; bu faktörler ikinci düzeylerinde çalıştırılmalıdır. Momentum katsayısı, öğrenme katsayısı, ÇKA iterasyon sayısı ve gizli katman sayısı faktörlerindeki artış ise test doğruluğunun düşmesine neden olacaktır. Dolayısıyla bu faktörler ilk düzeylerinde çalıştırılmalıdır. Üç düzeye sahip olan işlem elemanı sayısı faktörünün birinci düzeyden ikinci düzeye çıkarılması test doğruluğunun düşmesine neden olacaktır, ancak ikinci düzeyden üçüncü düzeye geçiş ise kuvvetli bir sıçrama ile büyük bir artışa neden olacaktır. Ana etki grafiğine göre bu faktörün üçüncü düzeyi en etkin sonucu verecektir. Faktörlerin etkinlik ölçütü üzerindeki etkilerine göre sıralamaları, ortalamalar ile birlikte Tablo 7.22.'de verilmektedir.

Tablo 7.22. Iris veri kümesi için Taguchi faktör sıralaması.

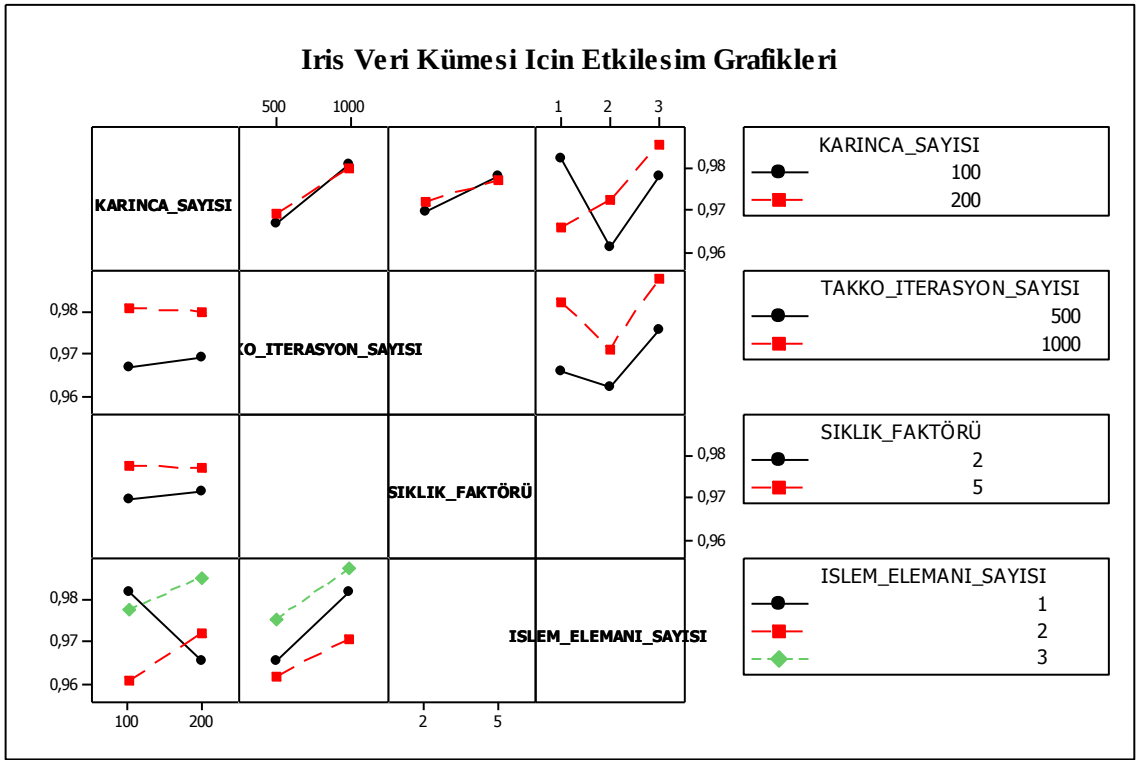
Faktörler Düzeyleri	M	T	f	Q	ρ	α	λ	Epoch	GK	İES
1	0,9737	0,9678	0,9707	0,9726	0,9726	0,98	0,9763	0,9785	0,9826	0,9739
2	0,9744	0,9804	0,9774	0,9756	0,9756	0,9681	0,9719	0,9696	0,9656	0,9667
3										0,9817
Fark	0,0007	0,0126	0,0067	0,003	0,003	0,0119	0,0044	0,0089	0,017	0,015
Sıralama	10	3	6	9	8	4	7	5	1	2

Tablo 7.22. ve Şekil 7.5.'den elde edilen bilgilere göre, bu veri kümesinde faktörlerin test doğruluğu üzerindeki etkilerine göre sıralaması ve en iyi sonuç veren düzeyleri Tablo 7.23.'de gösterilmektedir. Karınca sayısı faktörü test doğruluğunu etkilemediği için tabloda yer almamaktadır.

Tablo 7.23. Iris veri kümesi için faktör sıralamaları ve düzeyleri.

Faktör	GK	İES	T	α	Epoch	f	λ	ρ	Q
Düzey	1	10	1000	0,7	10000	5	0,2	0,9	10

Şekil 7.6. Iris veri kümesi için oluşturulan ikili etkileşim grafiklerini göstermektedir. Şekilden de görüldüğü gibi sadece karınca sayısı "M" ile gizli katman(lar)daki işlem elemanı sayısı "İES" arasında kuvvetli bir ikili etkileşim vardır. O kadar kuvvetli olmasa da karınca sayısı ile TAKKO iterasyon sayısı "T" arasında ve yine karınca sayısı ile sıklık faktörü "f" arasında bir etkileşim olduğu söylenebilir. Dolayısıyla ana etkilere göre bu faktörlerin test doğruluğunu etkilediği kesin olarak söylenemez. Etkileşimleri de dikkate alarak karar vermek gerekir.



Şekil 7.6. Iris veri kümesi etkileşim grafiği.

Ana etkiler ile birlikte ikili etkileşimler dikkate alındığında; ana etkilere göre etkisiz çıkan karınca sayısı faktörü ikili etkileşimlere sahiptir. Dolayısı ile bu faktörün aldığı düzey test doğruluğunu etkileyebilir. Şekil 7.6.'da verilen etkileşim eğrilerine göre eğer bu faktör ikinci düzeyinde çalıştırılırsa test doğruluğunda bir artışa neden olur sonucuna varılabilir.

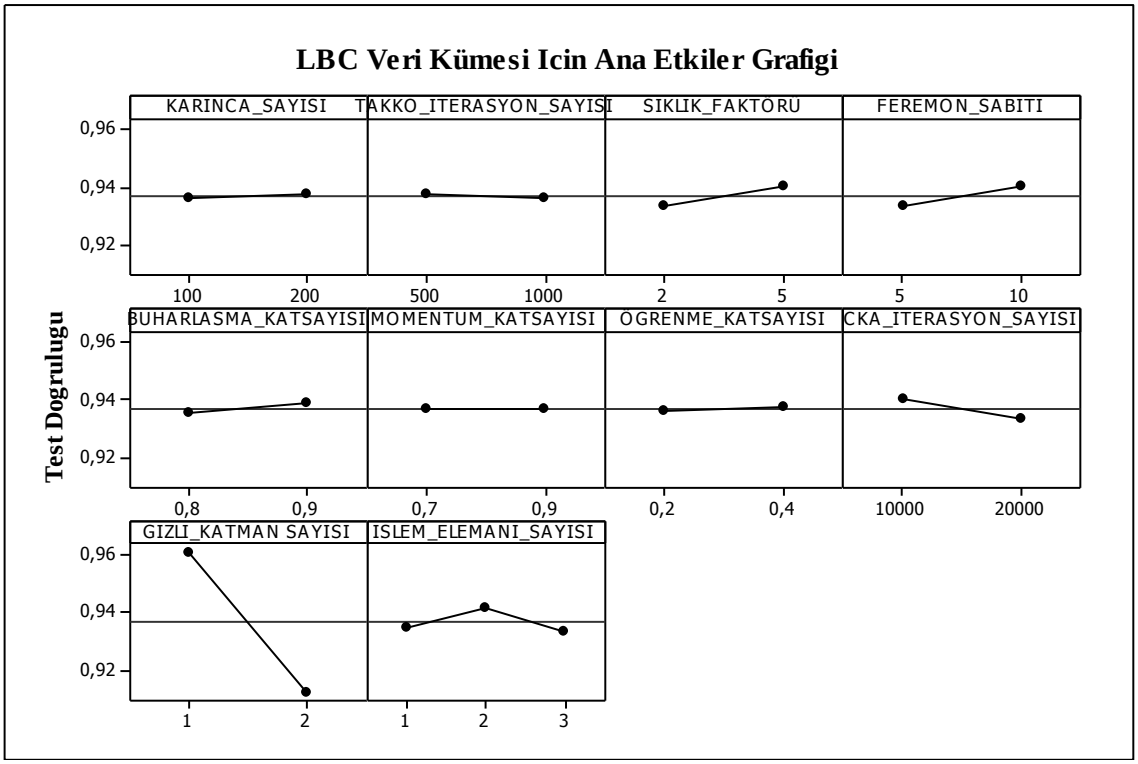
Geliştirilen algoritmada Iris veri kümesi için kullanılan faktör düzeyleri Tablo 7.24.'de verilmektedir.

Tablo 7.24. Iris veri kümesi için belirlenen faktör düzeyleri.

Faktör	M	T	f	Q	ρ	α	λ	Epoch	GK	İES
Değer	200	1000	5	10	0,9	0,7	0,2	10000	1	10

7.4.2.4. LBC Veri Kümesi Deney Tasarımı

LBC veri kümesi için hazırlanan Taguchi deney tasarımının analizi ile elde edilen ana etkiler grafiği Şekil 7.7.'de gösterilmektedir.



Şekil 7.7. LBC veri kümesi ana etkiler grafiği.

Şekildeki ana etki grafiklerine göre, karınca sayısı, TAKKO iterasyon sayısı, momentum katsayısı ve öğrenme katsayısı faktörlerinin test doğruluğu üzerinde etkisi yoktur ya da çok az etkisi vardır. Sıklık faktörü ve buharlaşma katsayısı faktörlerinin

etkinlik ölçütü üzerinde pozitif etkisi olduğu yani bu faktörlerin düzeylerinin ilk değerinden ikinci değerine çıkarılmasının test doğruluğunu artıracağı söylenebilir. ÇKA iterasyon sayısı ve gizli katman sayısı ise test doğruluğu negatif yönde etkilemektedir, bu faktörlerin ilk değerlerinin kullanılması etkinlik ölçütünde daha iyi sonuçlar elde edilmesini sağlayacaktır. Üç düzeye sahip olan İES faktörünün ise farklı düzeyleri T.D.'yi farklı yönde etkilemektedir. Şekilden de görülebileceği gibi bu faktör ikinci düzeyinde çalıştırılırsa en iyi T.D. değeri elde edilebilir.

LBC veri kümesi için Taguchi deney tasarımı sonucunda elde edilen, faktörlerin etkinlik ölçütü üzerindeki etkilerine göre sıralamaları, ortalamalar ile birlikte Tablo 7.25.'de verilmektedir.

Tablo 7.25. LBC veri kümesi için Taguchi faktör sıralaması.

Faktörler / Düzeyleri	M	T	f	Q	ρ	α	λ	Epoch	GK	İES
1	0,9361	0,9373	0,9332	0,9331	0,935	0,9368	0,9359	0,9402	0,9607	0,9346
2	0,9373	0,9361	0,9402	0,9403	0,9383	0,9365	0,9375	0,9332	0,9127	0,9416
3										0,9338
Fark	0,0012	0,0012	0,0071	0,0072	0,0033	0,0003	0,0016	0,007	0,0481	0,0078
Sıralama	9	8	4	3	6	10	7	5	1	2

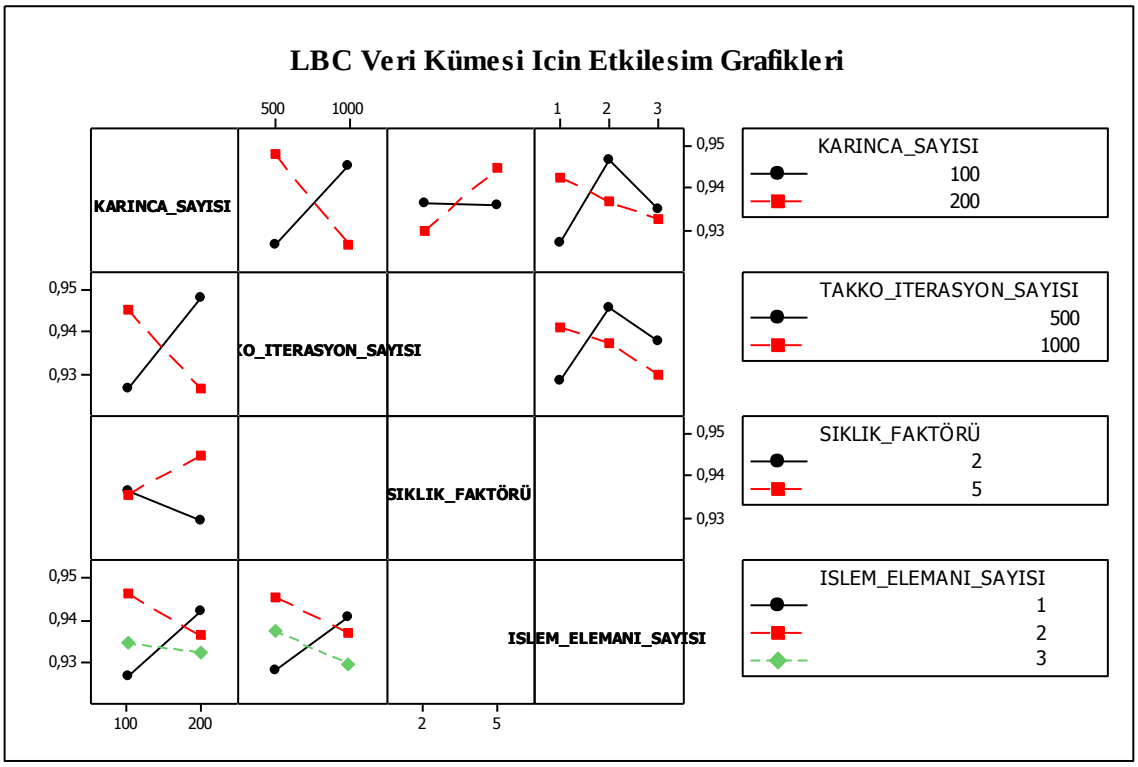
Tablo 7.25. ve Şekil 7.7.'den elde edilen bilgilere göre, bu veri kümesinde faktörlerin test doğruluğu üzerindeki etkilerine göre sıralaması ve en iyi sonuç veren düzeyleri Tablo 7.26.'da gösterilmektedir. Etkisiz faktörlere tabloda yer verilmemiştir.

Tablo 7.26. LBC veri kümesi için faktör sıralamaları ve düzeyleri.

Faktör	f	Q	ρ	Epoch	GK	İES
Düzey	5	10	0,9	10000	1	11

Şekil 7.8. ikili etkileşim grafiklerini göstermektedir. Karınca sayısı-TAKKO iterasyon sayısı, karınca sayısı-sıklık faktörü, karınca sayısı-işlem elemanı sayısı ve TAKKO iterasyon sayısı-işlem elemanı sayısı faktörleri arasında kuvvetli ikili etkileşimler

olduğu şekilden görülmektedir. Dolayısıyla etkin faktör düzeylerine karar verirken bu etkileşimleri de dikkate almak gerekir.



Şekil 7.8. LBC veri kümesi etkileşim grafiği.

Sonuç olarak, ana etkiler ile birlikte ikili etkileşimler dikkate alındığında; ana etkilere göre etkisiz çıkan karınca sayısı ve TAKKO iterasyon sayısı faktörleri kuvvetli ikili etkileşimlere sahiptir. Bu nedenle bu iki faktörün düzeyi test doğruluğunu etkileyebilir. İkili etkileşim eğrilerine göre karınca sayısı ikinci düzeyinde, TAKKO iterasyon sayısı ise ilk düzeyinde çalıştırılırsa test doğruluğunda artış sağlanabilir. İkili etkileşimler, ana etkiler sonucu belirlenen sıklık faktörü ve işlem elemanı sayısı faktörlerinin düzeylerinde değişiklik yaratmamıştır. Bu faktörler için ikili etkileşimlerin ana etkileri kuvvetlendirdiği söylenebilir.

Ana etkiler sonucu etkisiz çıkan momentum katsayısı ve öğrenme katsayısı faktörlerinde herhangi bir etkileşim çıkmamıştır. Dolayısıyla bu faktörlerin hangi düzeyde çalıştırıldığının test doğruluğu üzerinde etkisi yoktur.

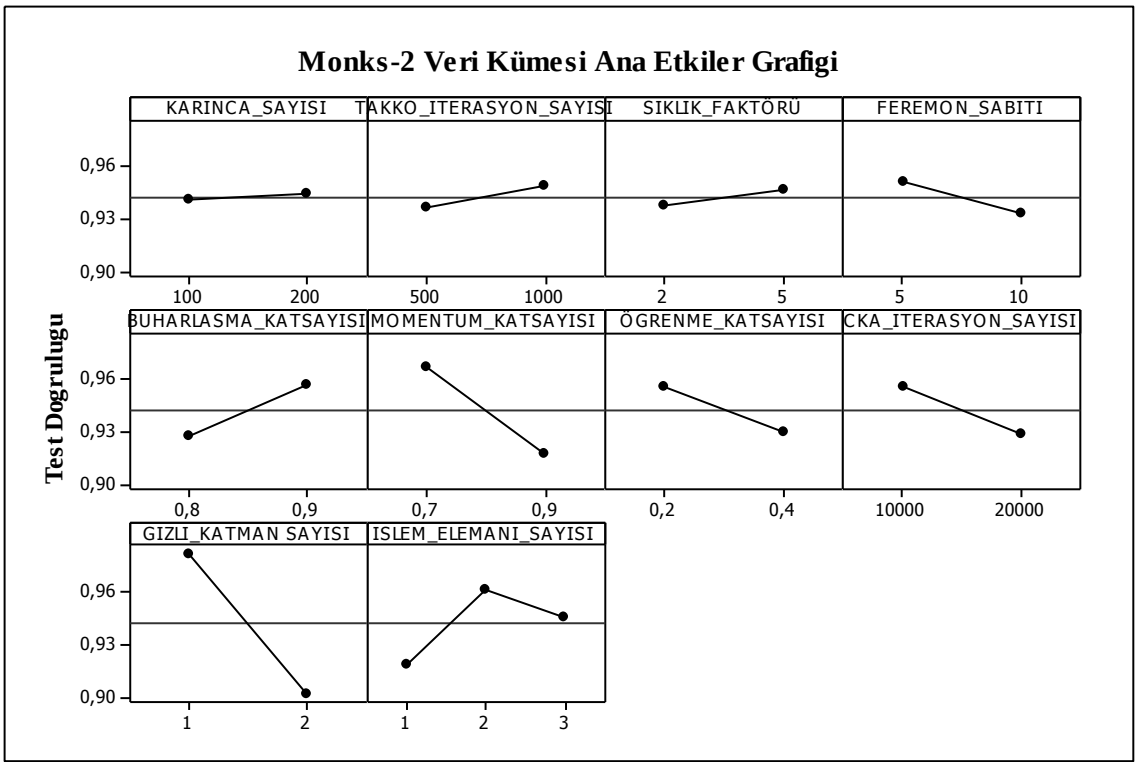
Geliştirilen algoritmada LBC veri kümesi için kullanılan faktör düzeyleri Tablo 7.27.'de verilmektedir.

Tablo 7.27. LBC veri kümesi için belirlenen faktör düzeyleri.

Faktör	M	T	f	Q	ρ	α	λ	Epoch	GK	İES
Değer	200	500	5	10	0,9	0,9	0,4	10000	1	11

7.4.2.5. Monks-2 Veri Kümesi Deney Tasarımı

Şekil 7.9. Monks-2 veri kümesi için hazırlanan Taguchi deney tasarımının analizi ile elde edilen ana etkiler grafiğini göstermektedir. Şekilden de görülebileceği gibi tüm faktörlerin test doğruluğu üzerinde etkisi vardır, ancak karınca sayısı faktörü en az etkiye sahip faktördür.



Şekil 7.9. Monks-2 veri kümesi ana etkiler grafiği.

TAKKO iterasyon sayısı, sıklık faktörü, buharlaşma katsayısı ve karınca sayısı faktörlerinin düzeylerindeki artış etkinlik ölçütünde artışa neden olacaktır yani bu faktörlerin etkisi pozitif yönlüdür. İşlem elemanı hariç kalan diğer faktörlerin ise etkileri test doğruluğu ile zıt yönlü olup düzeylerdeki artış doğruluğu azaltacak yöndedir. İşlem elemanı faktörünün düzeylerinde ise iki durum da mevcuttur. Birinci düzeyden ikinci düzeye çıkış test doğruluğunu artırırken ikinci düzeyden üçüncü düzeye geçiş test doğruluğunu azaltan yöndedir.

Monks-2 için Taguchi deney tasarımı sonucunda elde edilen, faktörlerin etkinlik ölçütü üzerindeki etkilerine göre sıralamaları, ortalamalar ile birlikte Tablo 7.28.'de verilmektedir.

Tablo 7.28. Monks-2 veri kümesi için Taguchi faktör sıralaması.

Faktörler	M	T	f	Q	ρ	α	λ	Epoch	GK	İES
Düzeyleri										
1	0,9401	0,9358	0,937	0,9505	0,9274	0,9662	0,9546	0,9554	0,9811	0,9187
2	0,9436	0,948	0,9468	0,9332	0,9564	0,9175	0,9291	0,9283	0,9027	0,9608
3										0,9461
Fark	0,0036	0,0122	0,0098	0,0173	0,029	0,0487	0,0255	0,0271	0,0784	0,0421
Sıralama	10	8	9	7	4	2	6	5	1	3

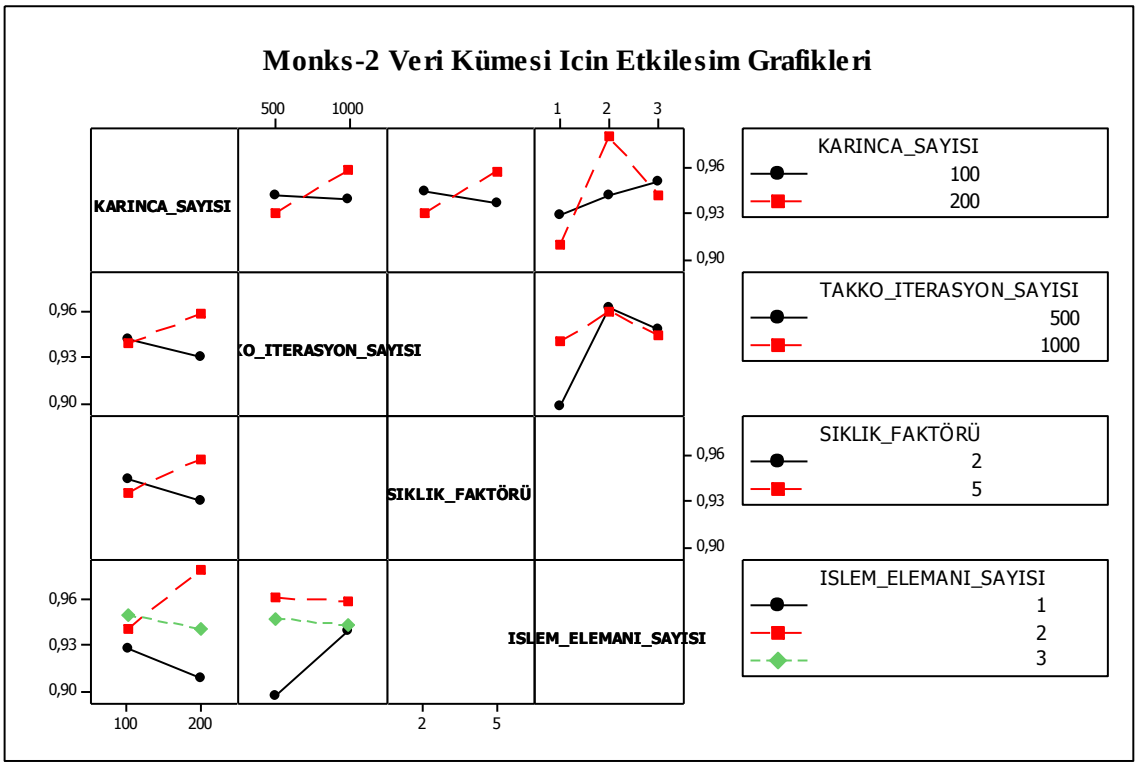
Tablo 7.28. ve Şekil 7.9.'dan elde edilen bilgilere göre, faktörlerin test doğruluğu üzerindeki etkilerine göre sıralaması ve en iyi sonuç veren düzeyleri Tablo 7.29.'da gösterilmektedir.

Tablo 7.29. Monks-2 veri kümesi için faktör sıralamaları ve düzeyleri.

Faktör	GK	α	İES	ρ	Epoch	λ	Q	T	f	M
Düzey	1	0,7	7	0,9	10000	0,2	5	1000	5	200

Şekil 7.10. Monks-2 problem için ikili etkileşim grafiklerini göstermektedir. Karınca sayısı faktörü ile TAKKO iterasyon sayısı, sıklık faktörü ve işlem elemanı sayısı faktörlerinin etkileşim eğrileri birbirini kesmektedir. Dolayısıyla bu faktörler arasında

kuvvetli ikili etkileşimler söz konusudur. TAKKO iterasyon sayısı faktörü ile işlem elemanı sayısı faktörleri arasında da zayıf bir etkileşim söz konusudur.



Şekil 7.10. Monks-2 veri kümesi etkileşim grafiği.

Sonuç olarak, ana etkiler ile birlikte ikili etkileşimler dikkate alındığında, faktörler arasında ortaya çıkan etkileşimler ana etkileri desteklemektedir. Hiçbir faktörün düzeyi etkileşimler sonucu değişmemiş, ana etkilerle belirlenen düzeylerinde kalmıştır. Bu nedenle etkileşimlerin ana etkileri kuvvetlendirdiği söylenebilir.

Geliştirilen algoritmada Monks-2 veri kümesi için kullanılan faktör düzeyleri Tablo 7.30.'da verilmektedir.

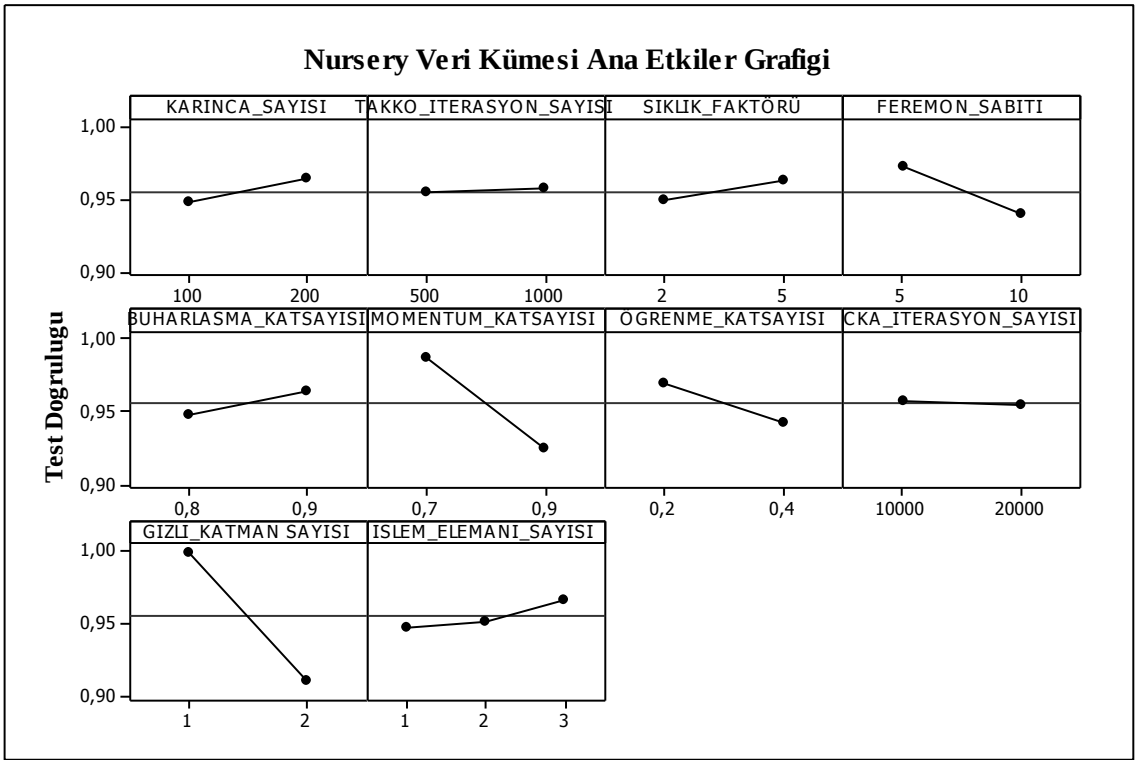
Tablo 7.30. Monks-2 veri kümesi için belirlenen faktör düzeyleri.

Faktör	M	T	f	Q	ρ	α	λ	Epoch	GK	İES
Değer	200	1000	5	5	0,9	0,7	0,2	10000	1	7

7.4.2.6. Nursery Veri Kümesi Deney Tasarımı

Nursery veri kümesi için hazırlanan Taguchi deney tasarımının analizi ile elde edilen ana etkiler grafiği Şekil 7.11.'de gösterilmektedir.

Şekildeki ana etki grafiklerine göre, çok katmanlı algılayıcı ve TAKKO iterasyon sayısı faktörlerinin (T ve Epoch) test doğruluğu üzerinde etkileri yok denecek kadar azdır. Karınca sayısı, sıklık faktörü, buharlaşma katsayısı ve işlem elemanı sayısı faktörlerinin etkinlik ölçütü üzerinde pozitif etkisi olduğu yani bu faktörlerin düzeylerinin ilk değerinden ikinci değerine çıkarılmasının test doğruluğunu artıracağı söylenebilir. Bu faktörler son düzeylerinde çalıştırılmalıdır. Kalan diğer parametreler için ise tam tersi bir durum söz konusudur. Parametre değerlerinin ikinci düzeylerine çıkarılması test doğruluğunu düşürecektir. Dolayısıyla kalan parametreler ilk düzeylerinde çalıştırılmalıdır. Böylece etkinlik ölçütü üzerinde en iyi sonuç elde edilebilir.



Şekil 7.11. Nursery veri kümesi ana etkiler grafiği.

Nursery veri kümesi için Taguchi deney tasarımı sonucunda elde edilen, faktörlerin etkinlik ölçütü üzerindeki etkilerine göre sıralamaları, ortalamalar ile birlikte Tablo

7.31.'de verilmektedir. TAKKO ve ÇKA iterasyon sayısı (T ve epoch) faktörleri ana etkilerde etkisiz çıktığı için tabloda yer almamaktadır.

Tablo 7.31. Nursery veri kümesi için Taguchi faktör sıralaması.

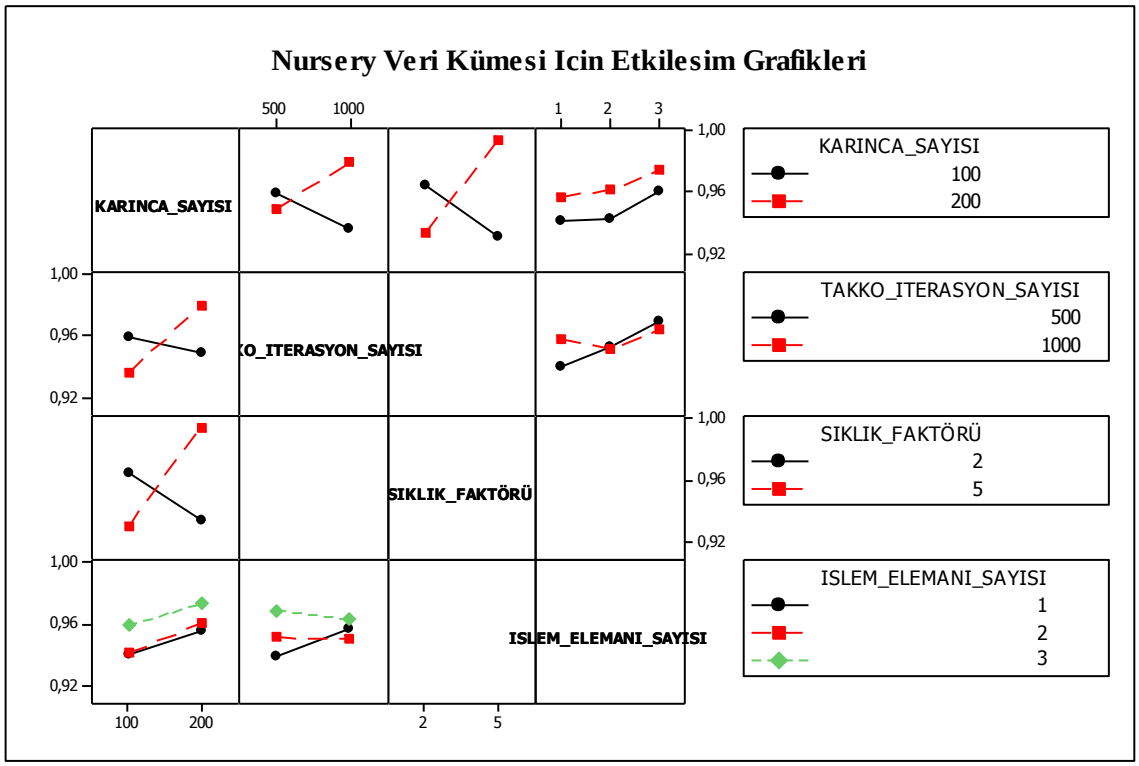
Faktörler	M	T	f	Q	ρ	α	λ	Epoch	GK	İES
Düzeyleri										
1	0,9473	0,9537	0,9491	0,9719	0,9473	0,9867	0,9687	0,9575	0,9997	0,9486
2	0,9639	0,9575	0,9621	0,9393	0,9639	0,9245	0,9425	0,9538	0,9115	0,9515
3										0,9668
Fark	0,0167	0,0038	0,013	0,0326	0,0166	0,0621	0,0261	0,0037	0,0882	0,0182
Sıralama	6	9	8	3	7	2	4	10	1	5

Tablo 7.31 ve Şekil 7.11.'den elde edilen bilgilere göre, bu veri kümesinde faktörlerin test doğruluğu üzerindeki etkilerine göre sıralaması ve en iyi sonuç veren düzeyleri Tablo 7.32.'de gösterilmektedir.

Tablo 7.32. Nursery veri kümesi için faktör sıralamaları ve düzeyleri.

Faktör	GK	α	Q	λ	İES	M	ρ	f
Düzey	1	0,7	5	0,2	21	200	0,9	5

Şekil 7.12. ikili etkileşim grafiklerini göstermektedir. Karınca sayısı-TAKKO iterasyon sayısı, karınca sayısı-sıklık faktörü ve TAKKO iterasyon sayısı-işlem elemanı sayısı faktörleri arasında ikili etkileşim mevcuttur. Karınca sayısı-gizli katman(lar)daki işlem elemanı sayısı arasındaki etkileşim ise yok denecek kadar azdır.



Şekil 7.12. Nursery veri kümesi etkileşim grafiği.

Sonuç olarak, ana etkiler ile birlikte ikili etkileşimler dikkate alındığında; faktörler arasında çıkan etkileşimler ana etkileri kuvvetlendirmektedir. Etkileşimler ana etkiler ile belirlenen faktör düzeylerinde herhangi bir değişikliğe neden olmamıştır. Yalnızca ana etkilerde etkisiz çıkan TAKKO iterasyon sayısı faktörü ikili etkileşimlerde etkin çıkmıştır ve işlem elemanı sayısı faktörünün aldığı değerden etkilenmektedir. Ana etkiler sonucu düzey belirlemeye gerek duyulmayan TAKKO iterasyon sayısı faktörünün etkileşimler nedeni ile ikinci düzey değerini alması gerektiği görülmektedir.

Ana etkilerde etkisiz çıkan bir diğer faktör epoch sayısıdır. Bu faktör, herhangi bir etkileşime de sahip değildir. Dolayısıyla alacağı düzey test doğruluğunu etkilemeyecektir. Basitlik açısından program çalıştırılırken, ilk düzeyi tercih edilmiştir.

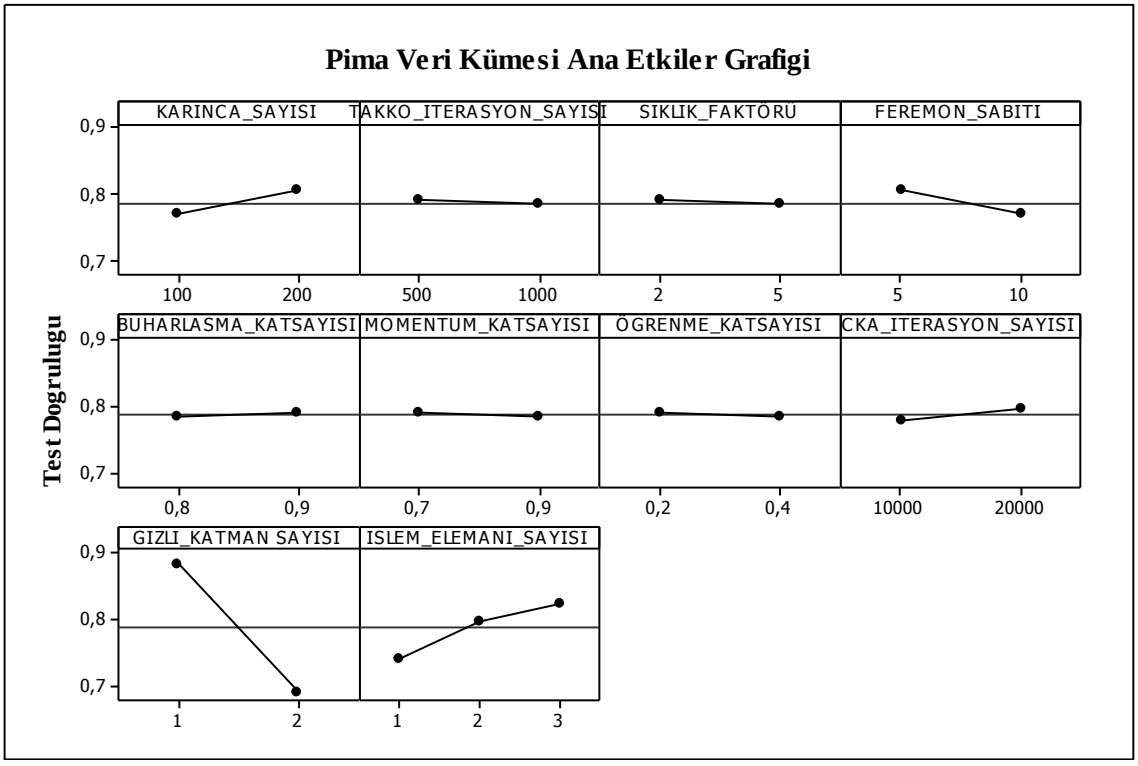
Geliştirilen algoritmada Nursery veri kümesi için kullanılan faktör düzeyleri Tablo 7.33.'de verilmektedir.

Tablo 7.33. Nursery veri kümesi için belirlenen faktör düzeyleri.

Faktör	M	T	f	Q	ρ	α	λ	Epoch	GK	İES
Değer	200	1000	5	5	0,9	0,7	0,2	10000	1	21

7.4.2.7. Pima Veri Kümesi Deney Tasarımı

Şekil 7.13. Pima veri kümesi için hazırlanan Taguchi deney tasarımının analizi ile elde edilen ana etkiler grafiğini göstermektedir.



Şekil 7.13. Pima veri kümesi ana etkiler grafiği.

Şekildeki ana etki grafiklerine göre, sadece karınca sayısı, feremon sabiti, gizli katman sayısı ve gizli katmanlardaki işlem elemanı sayısı faktörlerinin test doğruluğu üzerinde pozitif veya negatif yönde etkisi vardır. İşlem elemanı sayısı ve karınca sayısı faktörlerinin düzey artışları test doğruluğunu iyeleştirecektir. Bunun aksine feremon sabiti ve gizli katman sayısı faktörlerindeki düzey artışı test doğruluğunda düşüşe neden olacaktır. Diğer faktörler ise ya etkisizdir ya da etkileri yok denilecek kadar azdır.

Pima veri kümesi için Taguchi deney tasarımı sonucunda elde edilen, faktörlerin etkinlik ölçütü üzerindeki etkilerine göre sıralamaları, ortalamalar ile birlikte Tablo 7.34.'de verilmektedir.

Tablo 7.34. Pima veri kümesi için Taguchi faktör sıralaması.

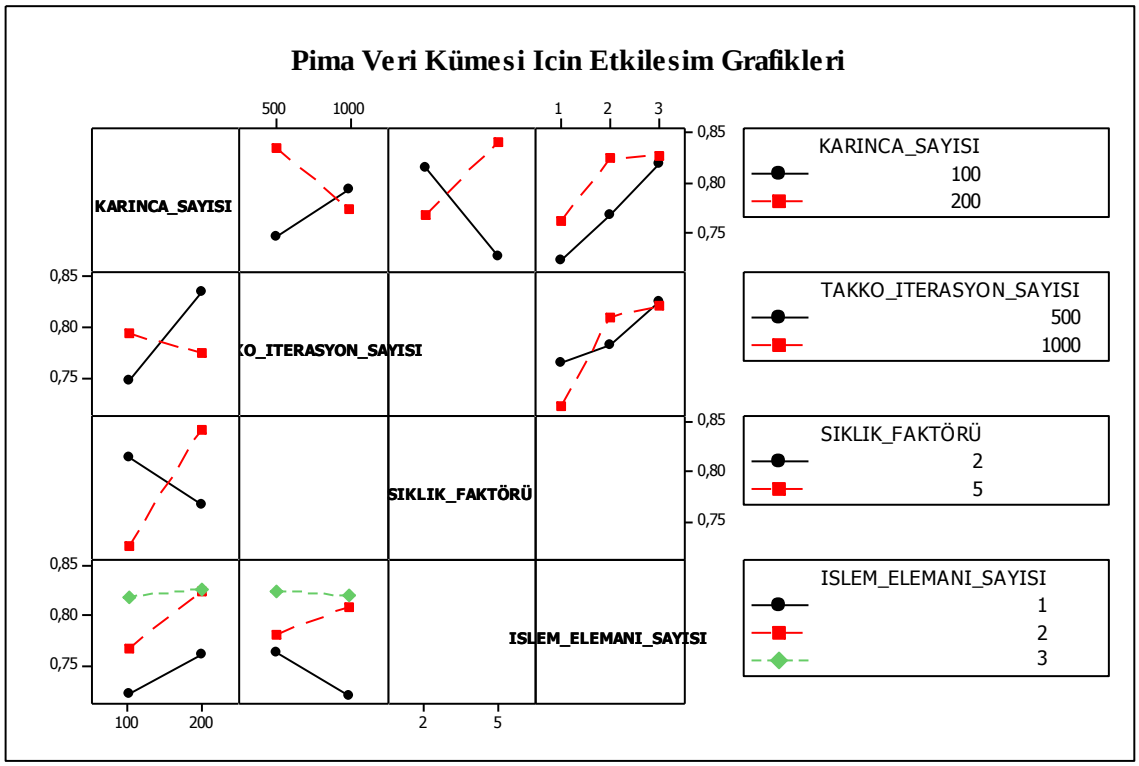
Faktörler / Düzeyleri	M	T	f	Q	ρ	α	λ	Epoch	GK	İES
1	0,7698	0,7909	0,7912	0,8058	0,7833	0,7897	0,7916	0,7777	0,8825	0,7418
2	0,8048	0,7837	0,7835	0,7688	0,7913	0,7849	0,783	0,7969	0,6921	0,7963
3										0,8238
Fark	0,0349	0,0071	0,0077	0,037	0,008	0,0048	0,0085	0,0192	0,1904	0,082
Sıralama	4	9	8	3	7	10	6	5	1	2

Tablo 7.34. ve Şekil 7.13.'den elde edilen bilgilere göre, bu veri kümesinde faktörlerin test doğruluğu üzerindeki etkilerine göre sıralaması ve en iyi sonuç veren düzeyleri Tablo 7.35.'de gösterilmektedir. Tabloda sadece ana etkilerde etkin için faktörlere yer verilmiştir.

Tablo 7.35. Pima veri kümesi için faktör sıralamaları ve düzeyleri.

Faktör	GK	İES	Q	M
Düzey	1	14	5	200

Şekil 7.14. ikili etkileşim grafiklerini göstermektedir. Karınca sayısı-TAKKO iterasyon sayısı, karınca sayısı-sıklık faktörü ve TAKKO iterasyon sayısı-gizli katman(lar)daki işlem elemanı sayısı etkileşim eğrileri birbirini kesmektedir. Dolayısıyla bu faktörler arasında ikili etkileşimler söz konusudur.



Şekil 7.14. Pima veri kümesi etkileşim grafiği.

Ana etkiler ile birlikte ikili etkileşimler dikkate alındığında; ikili etkileşimler karınca sayısı faktörünün düzeyinde bir değişikliğe neden olmamakta, etkiyi kuvvetlendirmektedir. Aynı durum gizli katman(lar)daki işlem elemanı faktörü için de geçerlidir.

Ana etkiler sonucu etkisiz çıkan TAKKO iterasyon sayısı ve sıklık faktörü ikili etkileşimlere sahiptir. Bu nedenle ana etkilere bakılarak bu faktörler için düzeyin önemi yoktur denilemez, etkileşimlere bakılarak karar vermek gerekir. Etkileşimlere göre TAKKO iterasyon sayısı faktörü ilk düzeyinde, sıklık faktörü ise ikinci düzeyinde çalıştırılırsa test doğruluğu artacaktır sonucuna varılabilir.

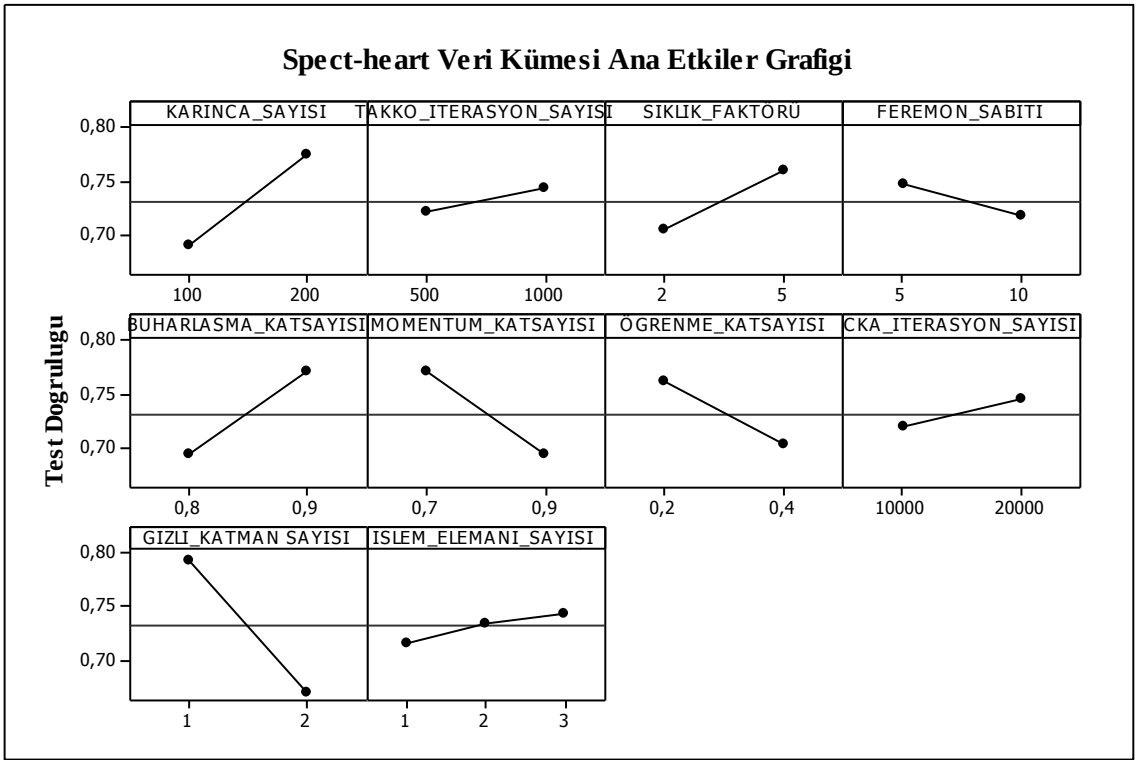
Geliştirilen algoritmada Pima veri kümesi için kullanılan faktör düzeyleri Tablo 7.36.'da verilmektedir.

Tablo 7.36. Pima veri kümesi için belirlenen faktör düzeyleri.

Faktör	M	T	f	Q	ρ	α	λ	Epoch	GK	İES
Değer	200	500	5	5	0,9	0,7	0,2	10000	1	14

7.4.2.8. Spect-heart Veri Kümesi Deney Tasarımı

Taguchi deney tasarımının analizi ile spect-heart veri kümesi için elde edilen ana etkiler grafiği Şekil 7.15.'de gösterilmektedir. Şekilden de görüldüğü gibi, bütün faktörlerin test doğruluğu üzerinde etkisi vardır. Karınca sayısı, TAKKO iterasyon sayısı, sıklık faktörü, buharlaşma katsayısı, ÇKA iterasyon sayısı ve işlem elemanı sayısı faktörlerinin etkinlik ölçütü üzerinde pozitif etkisi vardır yani bu faktörlerin düzeylerinin ilk değerinden ikinci veya üçüncü değerine çıkarılması test doğruluğunu artıracaktır. Kalan diğer parametreler için ise tam tersi bir durum söz konusudur. Parametre değerlerinin ikinci düzeylerine çıkarılması test doğruluğunu düşürecektir; dolayısıyla kalan parametreler ilk düzeylerinde çalıştırılmalıdır. Böylece etkinlik ölçütü üzerinde en iyi sonuç elde edilebilir.



Şekil 7.15. Spect-heart veri kümesi ana etkiler grafiği.

Spect-heart veri kümesi için Taguchi deney tasarımı sonucunda elde edilen faktörlerin etkinlik ölçütü üzerindeki etkilerine göre sıralamaları, ortalamalar ile birlikte Tablo 7.37.'de verilmektedir.

Tablo 7.37. Spect-heart veri kümesi için Taguchi faktör sıralaması.

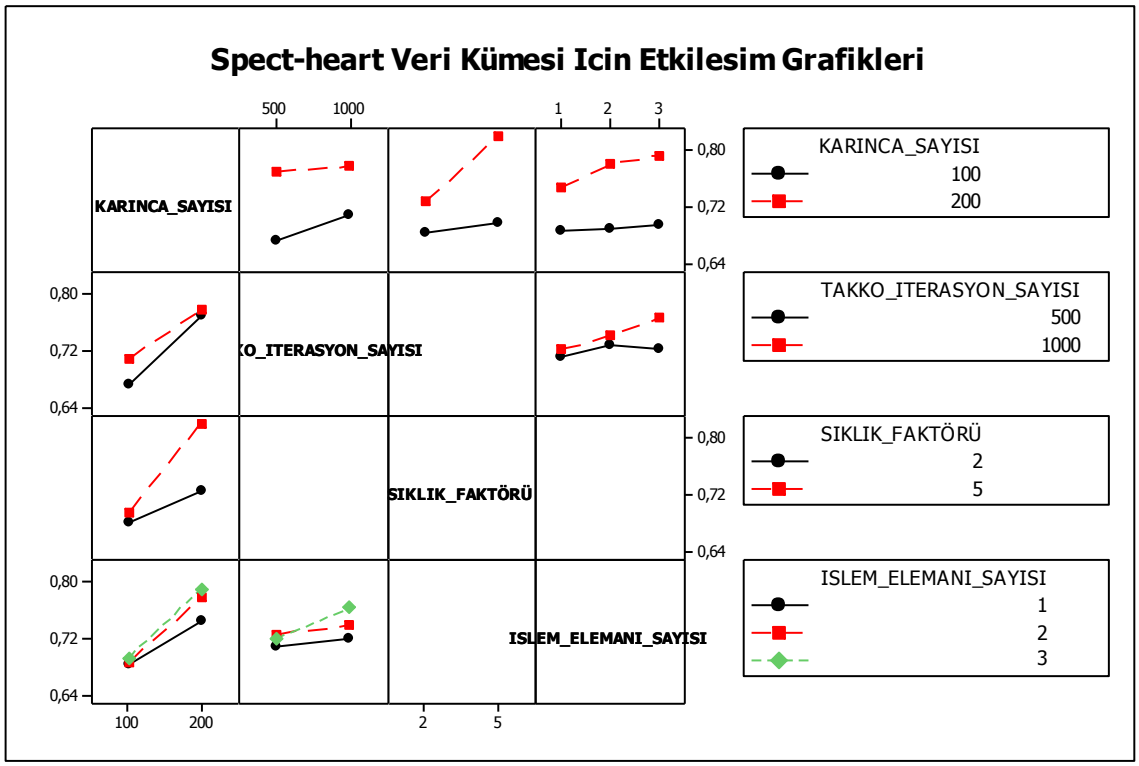
Faktörler Düzeyleri	M	T	f	Q	ρ	α	λ	Epoch	GK	İES
1	0,6898	0,7209	0,7048	0,7459	0,6925	0,7702	0,761	0,7191	0,7925	0,7166
2	0,7736	0,7425	0,7587	0,7176	0,771	0,6933	0,7025	0,7443	0,671	0,7348
3										0,7439
Fark	0,0838	0,0216	0,0539	0,0283	0,0785	0,0769	0,0585	0,0252	0,1216	0,0273
Sıralama	2	10	6	7	3	4	5	9	1	8

Tablo 7.37 ve Şekil 7.15.'den elde edilen bilgilere göre, bu veri kümesinde faktörlerin test doğruluğu üzerindeki etkilerine göre sıralaması ve en iyi sonuç veren düzeyleri Tablo 7.38.'de gösterilmektedir.

Tablo 7.38. Spect-heart veri kümesi için faktör sıralamaları ve düzeyleri.

Faktör	GK	M	ρ	α	λ	f	İES	Q	Epoch	T
Düzey	1	200	0,9	0,7	0,2	5	17	5	20000	1000

Şekil 7.16.'da etkileşim grafikleri gösterilmektedir. Şekilden görüldüğü gibi her bir etkileşim eğrisi yaklaşık olarak birbirine paraleldir. Dolayısıyla bu faktörler arasında ikili etkileşim söz konusu değildir.



Şekil 7.16. Spect-heart veri kümesi etkileşim grafiği.

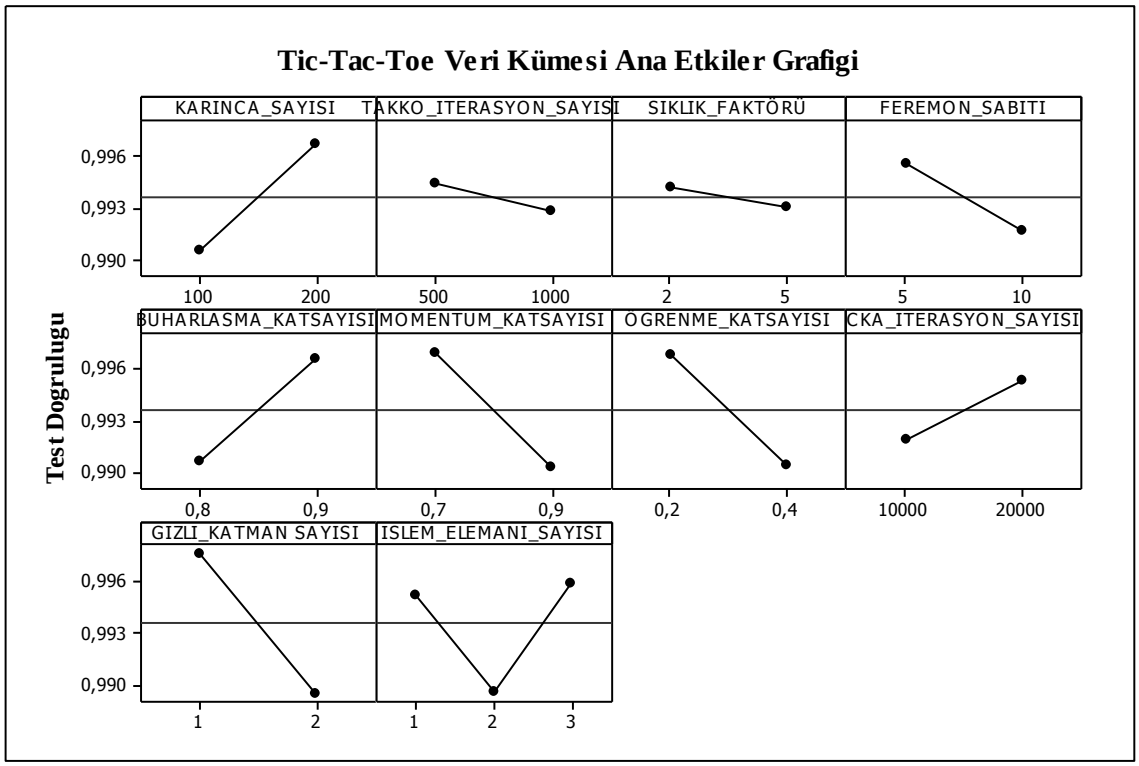
Geliştirilen algoritmada Spect-heart veri kümesi için kullanılan faktör düzeyleri Tablo 7.39.'da verilmektedir.

Tablo 7.39. Spect-heart veri kümesi için belirlenen faktör düzeyleri.

Faktör	M	T	f	Q	ρ	α	λ	Epoch	GK	İES
Değer	200	1000	5	5	0,9	0,7	0,2	20000	1	17

7.4.2.9. Tic-Tac-Toe Veri Kümesi Deney Tasarımı

Şekil 7.17.'de Tic-Tac-Toe veri kümesi için hazırlanan Taguchi deney tasarımının analizi ile elde edilen ana etkiler grafiği gösterilmektedir. Şekildeki ana etki grafiklerine göre, bütün faktörlerin test doğruluğu üzerinde aynı veya ters yönde etkisi vardır.



Şekil 7.17. Tic-Tac-Toe veri kümesi ana etkiler grafiği.

Bu faktörlerden karınca sayısı, buharlaşma katsayısı ve epoch sayısı test doğruluğunu pozitif etkilemekte, faktör düzeylerindeki artış test doğruluğunu da artırmaktadır. İşlem elemanı sayısı dışındaki diğer faktörler için ise tersi durum söz konusudur. Faktör düzeylerindeki artış test doğruluğunu düşürmektedir. İşlem elemanı sayısı faktörünün ise farklı düzeyleri etkinlik ölçütünü farklı yönde etkilemektedir. Birinci düzeyden ikinci düzeye geçiş test doğruluğunu azaltan yönde sonuç verirken; ikinci düzeyden üçüncü düzeye geçiş ise test doğruluğunu artıran yönde sonuç vermektedir.

Tic-Tac-Toe veri kümesi için Taguchi deney tasarımı sonucunda elde edilen, faktörlerin etkinlik ölçütü üzerindeki etkilerine göre sıralamaları, ortalamalar ile birlikte Tablo 7.40.'da verilmektedir.

Tablo 7.40. Tic-Tac-Toe veri kümesi için Taguchi faktör sıralaması.

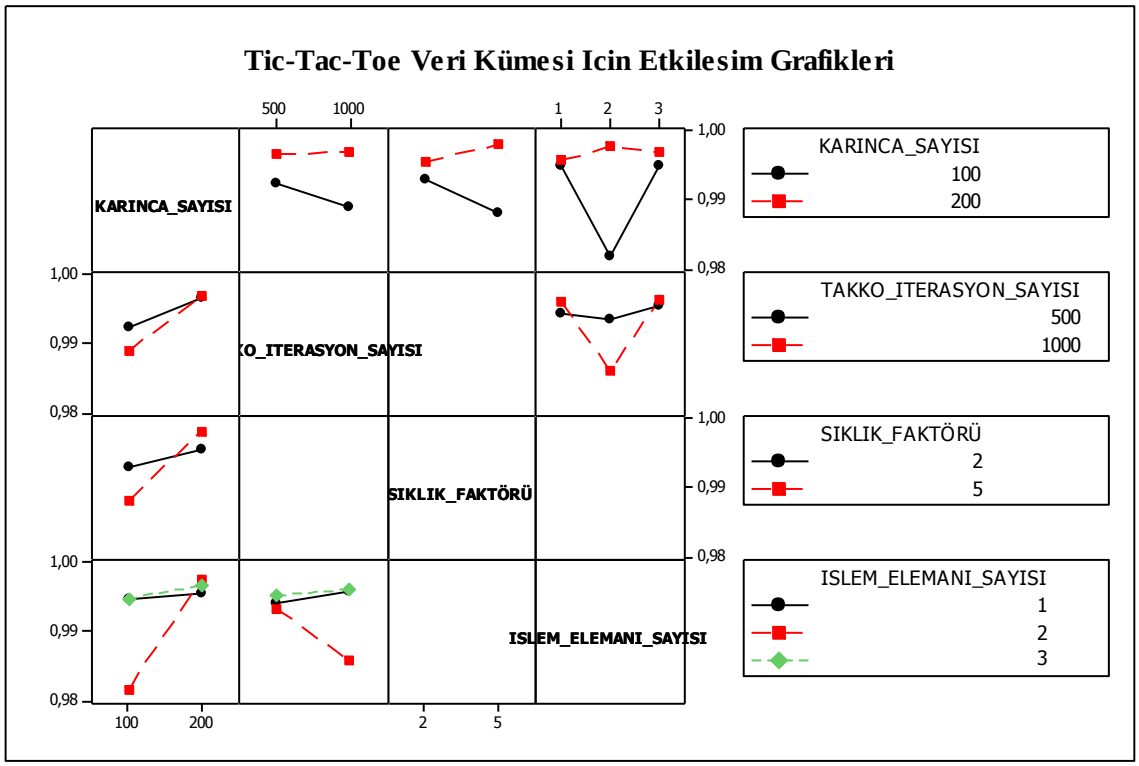
Faktörler Düzeyleri	M	T	f	Q	ρ	α	λ	Epoch	GK	İES
1	0,9905	0,9944	0,9942	0,9955	0,9906	0,997	0,9969	0,9919	0,9976	0,9952
2	0,9967	0,9928	0,993	0,9917	0,9966	0,9903	0,9904	0,9954	0,9896	0,9897
3										0,9959
Fark	0,0062	0,0016	0,0011	0,0038	0,006	0,0067	0,0065	0,0035	0,008	0,0062
Sıralama	4	9	10	7	6	2	3	8	1	5

Tablo 7.40 ve Şekil 7.17.'den elde edilen bilgilere göre, bu veri kümesinde faktörlerin test doğruluğu üzerindeki etkilerine göre sıralaması ve en iyi sonuç veren düzeyleri Tablo 7.41.'de gösterilmektedir.

Tablo 7.41. Tic-Tac-Toe veri kümesi için faktör sıralamaları ve düzeyleri.

Faktör	GK	α	λ	M	İES	ρ	Q	Epoch	T	f
Düzey	1	0,7	0,2	200	13	0,9	5	20000	500	2

Şekil 7.18. ikili etkileşim grafiklerini göstermektedir. Karınca sayısı-sıklık faktörü ve TAKKO iterasyon sayısı-gizli katman(lar)daki işlem elemanı sayısı etkileşim eğrileri birbirini kesmektedir. Dolayısıyla bu faktörler arasında ikili etkileşim söz konusudur.



Şekil 7.18. Tic-Tac-Toe veri kümesi etkileşim grafiği.

Sonuç olarak, ana etkiler ile birlikte ikili etkileşimler dikkate alındığında; ana etkilere göre ilk düzeyinde daha etkin olan TAKKO iterasyon sayısı faktörü "T", ikinci düzeyinde çalıştırırsa test doğruluğu artacaktır. İkili etkileşimler bu faktörün ana etkiler ile belirlenen düzeyinde değişikliğe neden olmuştur.

Benzer şekilde ana etki grafiğine göre ilk düzeyinde etkin çıkan sıklık faktörü değişkeninin etkinlik düzeyi etkileşimlerden etkilenmektedir. Ana etkilere göre ilk düzeyde çalıştırılması gereken sıklık faktörünün, etkileşimler sonucunda ikinci düzeyinde çalıştırılması gerektiği sonucuna ulaşılmaktadır. Böylece etkinlik ölçütünde iyileşme sağlanacaktır.

Etkileşime sahip olan karınca sayısı ve gizli katman(lar)daki işlem elemanı sayısı faktörlerinin düzeylerinde ise etkileşimden dolayı herhangi bir değişiklik olmayacaktır. Ana etkilerle belirlenen düzeyler korunacaktır.

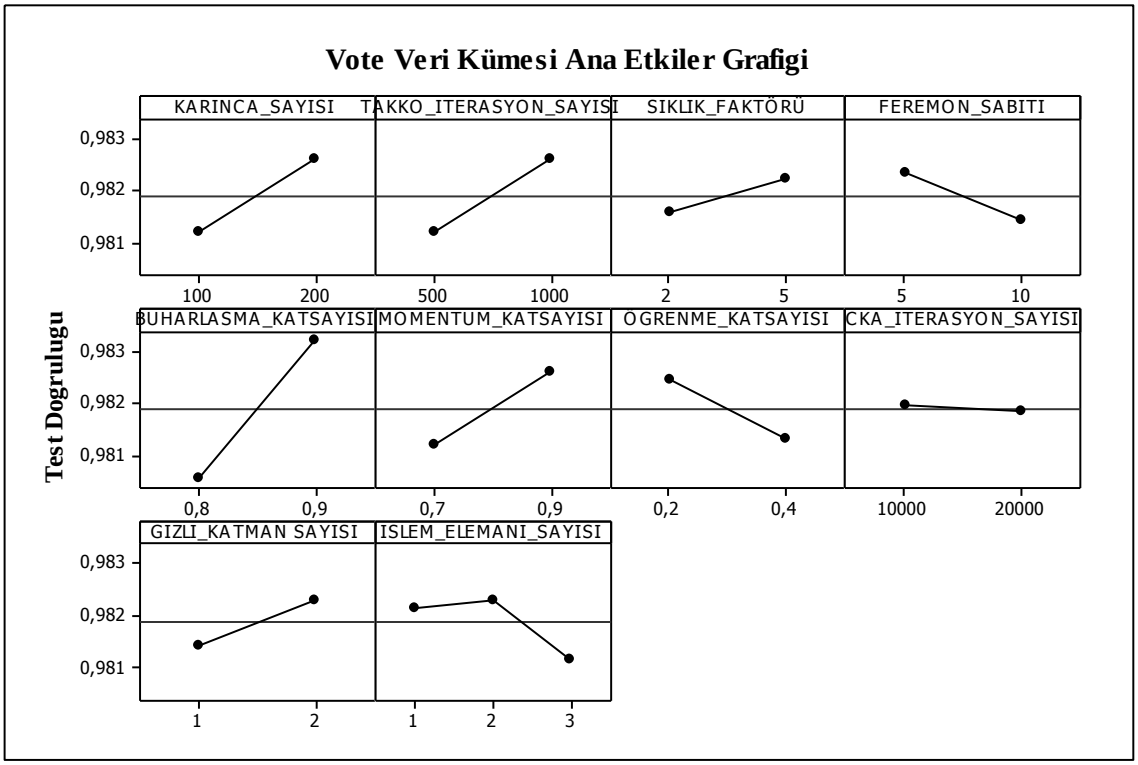
Geliştirilen algoritmada Tic-Tac-Toe veri kümesi için kullanılan faktör düzeyleri Tablo 7.42.'de verilmektedir.

Tablo 7.42. Tic-Tac-Toe veri kümesi için belirlenen faktör düzeyleri.

Faktör	M	T	f	Q	ρ	α	λ	Epoch	GK	İES
Değer	200	1000	5	5	0,9	0,7	0,2	20000	1	13

7.4.2.10. Vote Veri Kümesi Deney Tasarımı

Vote veri kümesi için hazırlanan Taguchi deney tasarımının analizi ile elde edilen ana etkiler grafiği Şekil 7.19.'da gösterilmektedir. Şekildeki ana etki grafiklerine göre, ÇKA iterasyon sayısı faktörü "Epoch" dışında kalan bütün faktörlerin test doğruluğu üzerinde etkisi vardır. Epoch faktörünün etkisiz olmasının nedeni ana etki eğrisinin x-eksenine paralel olmasıdır.



Şekil 7.19. Vote veri kümesi ana etkiler grafiği.

Ana etki grafiklerine göre karınca sayısı, TAKKO iterasyon sayısı, sıklık faktörü, buharlaşma katsayısı, momentum katsayısı, gizli katman sayısı faktörleri test doğruluğunu pozitif yönde etkilemekte ve bu faktörlerin düzeylerindeki artış test

doğruluğunda artışa neden olmaktadır. Kalan diğer faktörler için ise tam tersi durum söz konusu olup, faktör düzeylerindeki artış test doğruluğunu düşürecektir. Dolayısıyla bu faktörler ilk düzeyleriyle çalıştırılmalıdır.

Veri kümesi için Taguchi deney tasarımı sonucunda elde edilen, faktörlerin etkinlik ölçütü üzerindeki etkilerine göre sıralamaları, ortalamalar ile birlikte Tablo 7.43.'de verilmektedir.

Tablo 7.43. Vote veri kümesi için Taguchi faktör sıralaması.

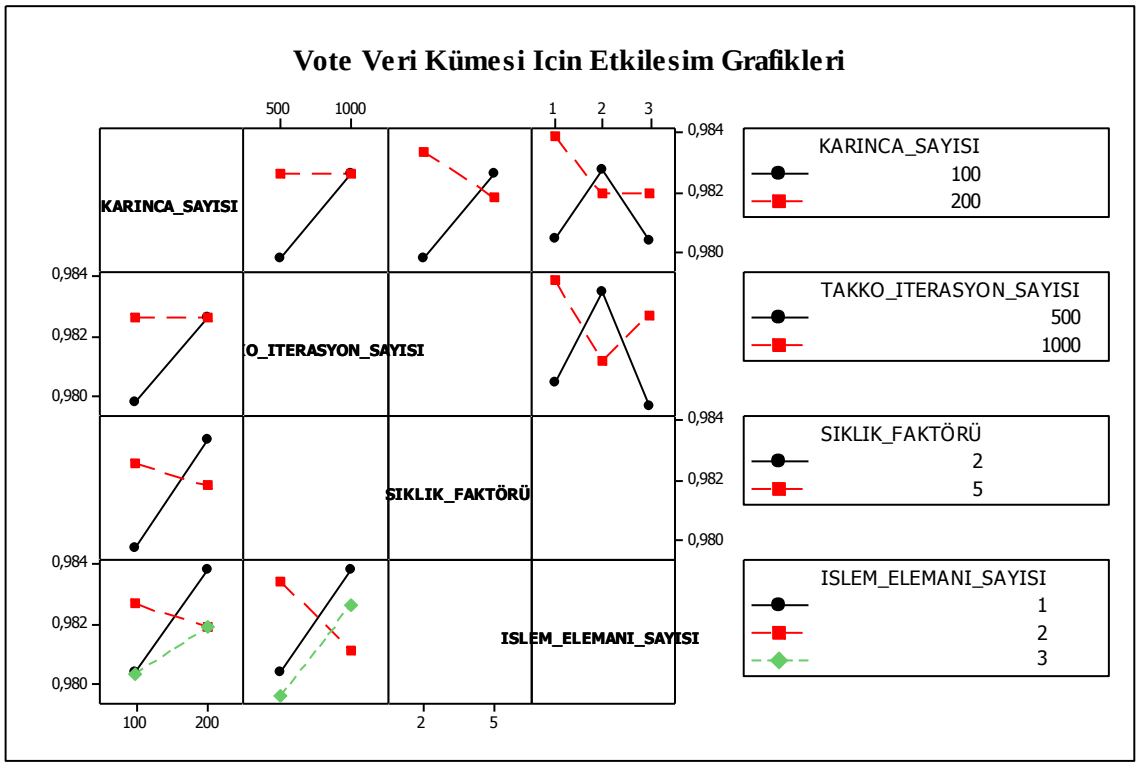
Faktörler / Düzeyleri	M	T	f	Q	ρ	α	λ	Epoch	GK	İES
1	0,9812	0,9812	0,9816	0,9823	0,9806	0,9812	0,9825	0,9819	0,9814	0,9821
2	0,9826	0,9826	0,9822	0,9814	0,9832	0,9826	0,9813	0,9818	0,9823	0,9823
3										0,9812
Fark	0,0014	0,0014	0,0006	0,0009	0,0027	0,0014	0,0012	0,0001	0,0009	0,0012
Sıralama	3	2	9	7	1	4	5	10	8	6

Elde edilen bu bilgilere göre, bu veri kümesinde faktörlerin test doğruluğu üzerindeki etkilerine göre sıralaması ve en iyi sonuç veren düzeyleri Tablo 7.44.'de gösterilmektedir.

Tablo 7.44. Vote veri kümesi için faktör sıralamaları ve düzeyleri.

Faktör	ρ	T	M	α	λ	İES	Q	GK	f
Düzey	0,9	1000	200	0,9	0,2	22/4	5	2	5

Şekil 7.20. ikili etkileşim grafiklerini göstermektedir. Karınca sayısı-sıklık faktörü, karınca sayısı-gizli katman(lar)daki işlem elemanı sayısı ve TAKKO iterasyon sayısı-gizli katman(lar)daki işlem elemanı sayısı etkileşim eğrileri birbirini kesmektedir. Dolayısıyla bu faktörler arasında ikili etkileşim söz konusudur.



Şekil 7.20. Vote veri kümesi etkileşim grafiği.

Sonuç olarak, ana etkiler ile birlikte ikili etkileşimler dikkate alındığında; ana etkilere göre ikinci düzeyinde daha etkin olan sıklık faktörü "f", ilk düzeyinde çalıştırırsa test doğruluğu artacaktır. İkili etkileşimler bu faktörün ana etkiler ile belirlenen düzeyinde değişikliğe neden olmuştur.

Benzer şekilde ana etki grafiğine göre gizli katman(lar)daki işlem elemanı sayısı "İES" faktörü ikinci düzeyinde çalıştırıldığında test doğruluğunda daha iyi çözüm elde edilmekteydi. İkili etkileşim grafiğinde bu faktörün TAKKO iterasyon sayısı faktörü ile arasındaki etkileşim eğrisi birinci düzeyinin test doğruluğu üzerinde artışa neden olacağını göstermektedir. Dolayısıyla İES faktörü ikinci düzeyinde çalıştırılmalıdır. Çok katmanlı algılayıcıdaki iterasyon sayısını gösteren "Epoch" parametresinin ise hangi düzeyde çalıştırıldığının test doğruluğu üzerinde etkisi yoktur. Ana etkilerde paralel bir doğruya sahiptir ve diğer faktörler ile arasında da etkileşim çıkmamıştır. Dolayısıyla geliştirilen algoritmada bu faktör zamandan tasarruf için ilk düzeyinde çalıştırılmıştır.

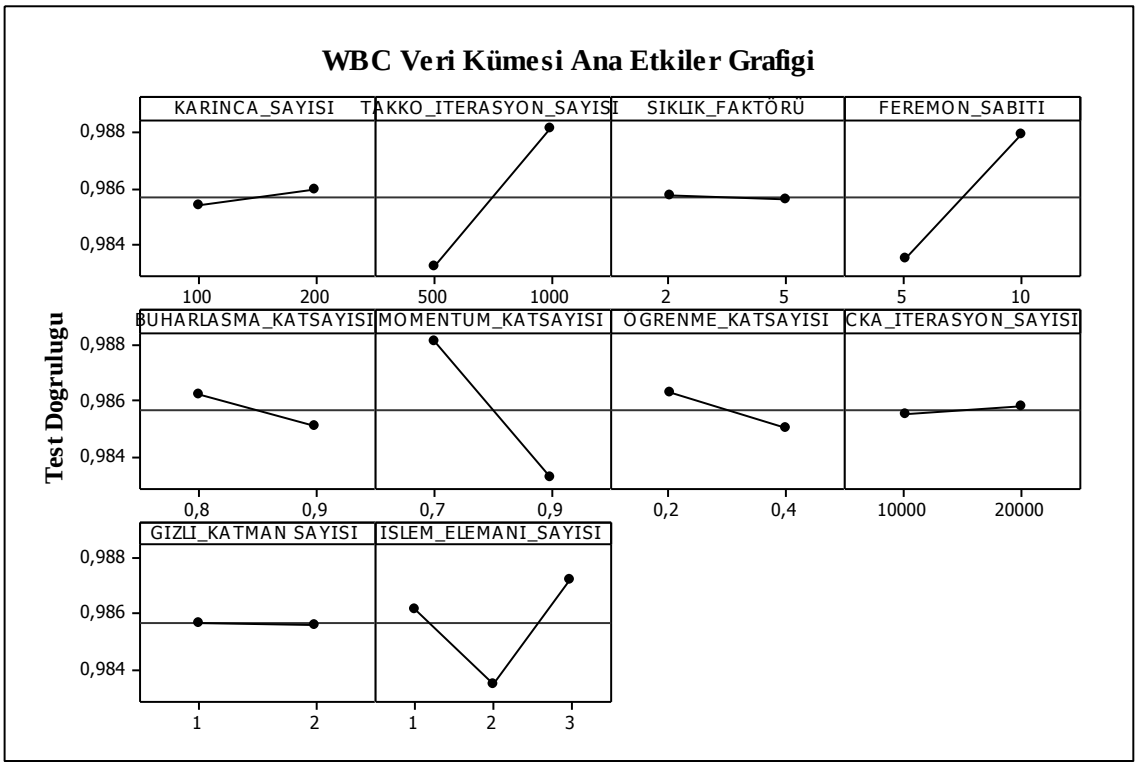
Geliştirilen algoritmada Vote veri kümesi için kullanılan faktör düzeyleri Tablo 7.45.'de verilmektedir.

Tablo 7.45. Vote veri kümesi için belirlenen faktör düzeyleri.

Faktör	M	T	f	Q	ρ	α	λ	Epoch	GK	İES
Değer	200	1000	2	5	0,9	0,9	0,2	10000	2	13/3

7.4.2.11. WBC Veri Kümesi Deney Tasarımı

Şekil 7.21. Heart-C veri kümesinde Taguchi tasarımı sonucu elde edilen ana etki grafiklerini göstermektedir. Şekildeki ana etki grafiklerine göre, sıklık faktörü, ÇKA iterasyon sayısı ve gizli katman sayısı faktörlerinin test doğruluğu üzerinde herhangi bir etkisi olmadığı söylenebilir, çünkü bu faktöre ait doğruların eğimi neredeyse sıfırdır. Bu üç faktör dışındaki diğer faktörlerin düzeyleri aynı yönde veya ters yönde test doğruluğunu etkilemektedir.



Şekil 7.21. WBC veri kümesi ana etkiler grafiği.

Karınca sayısı, TAKKO iterasyon sayısı ve feremon sabiti faktörlerinin düzeylerindeki artış test doğruluğunda da artışa neden olacaktır, dolayısıyla bu faktörlerin etkisi pozitif

yönlüdür denilebilir. Buharlaşma katsayısı, momentum katsayısı ve öğrenme katsayısı faktörlerinin test doğruluğu üzerindeki etkileri ise negatif yöndedir, yani düzeydeki artış test doğruluğunu düşürecektir. Üç düzeye sahip olan işlem elemanı sayısı faktörünün ise farklı düzeyleri test doğruluğunu farklı yönlerde etkilemektedir. En etkin düzeyi ise üçüncü düzeyidir denilebilir.

WBC veri kümesi için Taguchi deney tasarımı sonucunda elde edilen faktörlerin etkinlik ölçütü üzerindeki etkilerine göre sıralamaları, ortalamalar ile birlikte Tablo 7.46'da verilmektedir.

Tablo 7.46. WBC veri kümesi için Taguchi faktör sıralaması.

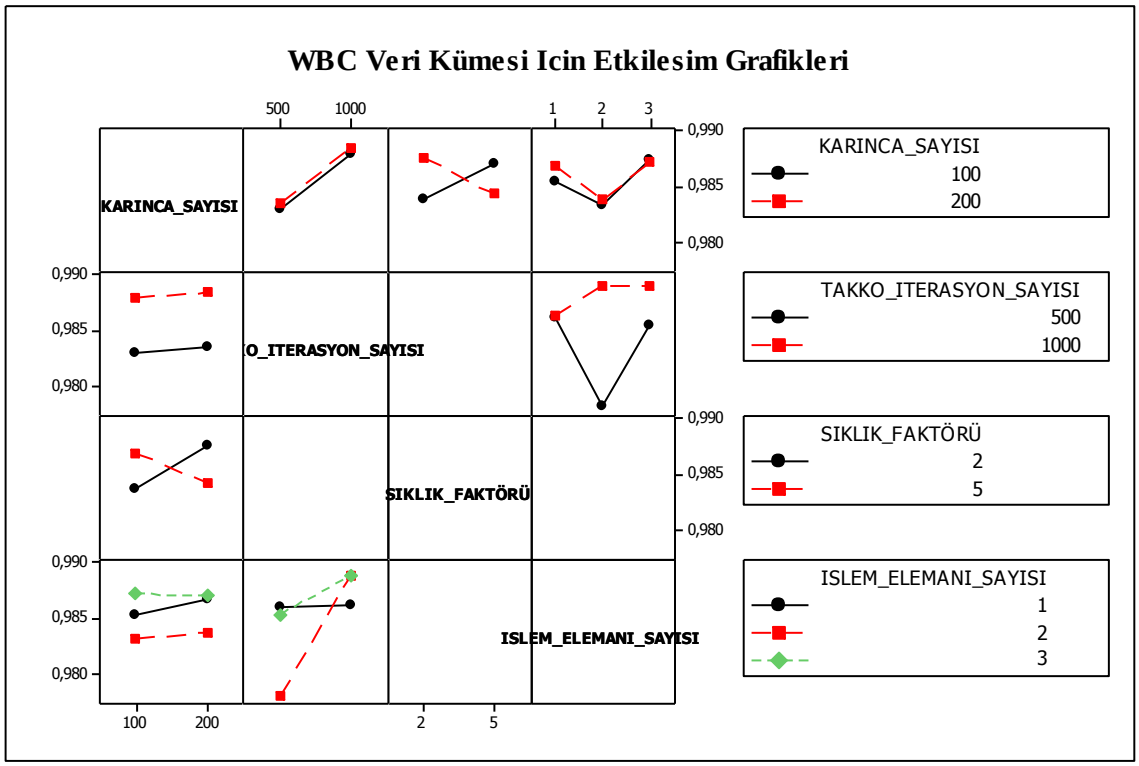
Faktörler Düzeyleri	M	T	f	Q	ρ	α	λ	Epoch	GK	İES
1	0,9854	0,9832	0,9857	0,9834	0,9862	0,9881	0,9863	0,9855	0,9857	0,9862
2	0,9859	0,9881	0,9856	0,9879	0,9851	0,9832	0,985	0,9858	0,9856	0,9836
3										0,9872
Fark	0,0005	0,0049	0,0001	0,0045	0,0011	0,0049	0,0013	0,0003	0,0001	0,0037
Sıralama	7	1	10	3	6	2	5	8	9	4

Tablo 7.46. ve Şekil 7.21.'den elde edilen bilgilere göre, bu veri kümesinde faktörlerin test doğruluğu üzerindeki etkilerine göre sıralaması ve en iyi sonuç veren düzeyleri Tablo 7.47.'de gösterilmektedir.

Tablo 7.47. WBC veri kümesi için faktör sıralamaları ve düzeyleri.

Faktör	T	α	Q	İES	λ	ρ	M
Düzey	1000	0,7	10	13	0,2	0,8	200

Şekil 7.22. WBC veri kümesi için oluşturulan ikili etkileşim grafiklerini göstermektedir. Karınca sayısı-sıklık faktörü ve TAKKO iterasyon sayısı-gizli katman(lar)daki işlem elemanı sayısı etkileşim eğrileri birbirini kesmektedir. Dolayısıyla bu faktörler arasında ikili etkileşim söz konusudur.



Şekil 7.22. WBC veri kümesi etkileşim grafiği.

Ana etkiler ile birlikte ikili etkileşimler dikkate alındığında; ana etkilere göre etkisiz çıkan sıklık faktörü "f", ilk düzeyinde çalıştırırsa test doğruluğu artacaktır. İkili etkileşimler bu faktörün ana etkiler ile belirlenen etkisizliğini ortadan kaldırmıştır.

Diğer etkileşime sahip faktörlerin düzeyleri ise ana etkilerle aynı yöndedir. Dolayısıyla bu faktörlerin ana etkileri kuvvetlendirdikleri sonucuna varılabilir. Çok katmanlı algılayıcıdaki iterasyon sayısını gösteren "Epoch" ve gizli katman sayısı faktörlerinin ise hangi düzeyde çalıştırıldığının test doğruluğu üzerinde etkisi yoktur. Ana etkilerde paralel bir doğruya sahiptirler ve diğer faktörler ile aralarında da etkileşim çıkmamıştır. Dolayısıyla geliştirilen algoritmada bu faktörler kolaylık açısından ilk düzeyinde çalıştırılmıştır.

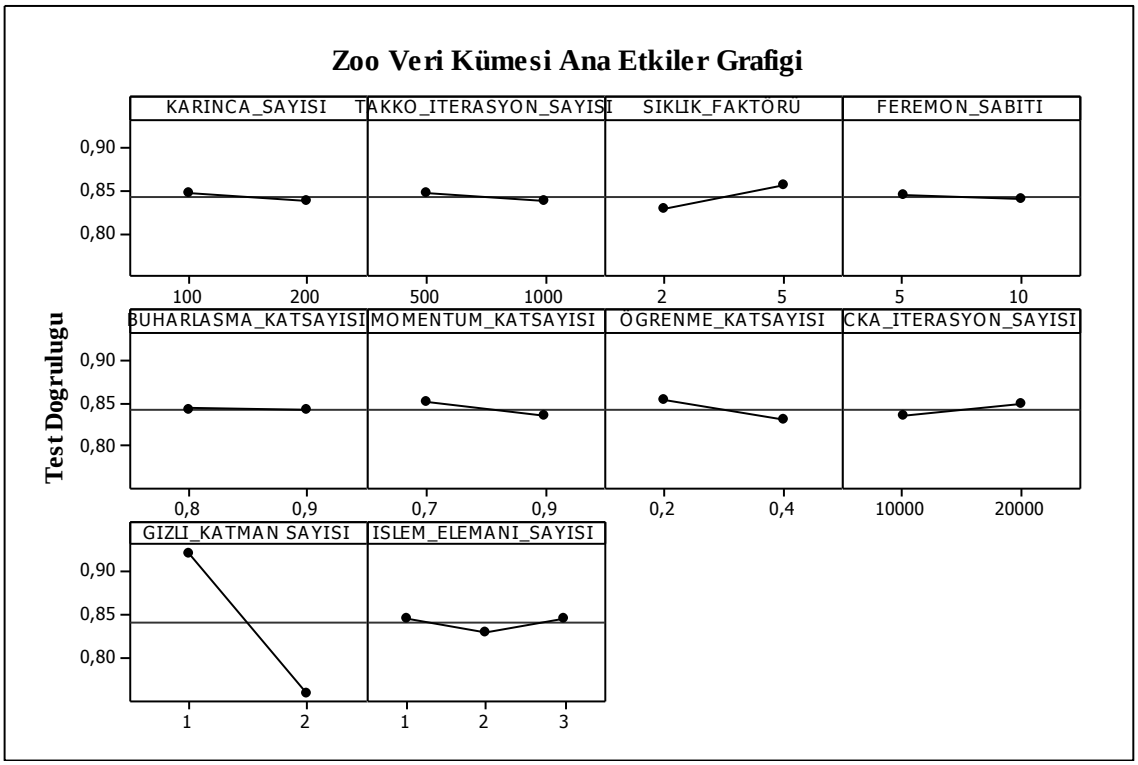
Geliştirilen algoritmada WBC veri kümesi için kullanılan faktör düzeyleri Tablo 7.48.'de verilmektedir.

Tablo 7.48. WBC veri kümesi için belirlenen faktör düzeyleri.

Faktör	M	T	f	Q	ρ	α	λ	Epoch	GK	İES
Değer	200	1000	2	10	0,8	0,7	0,2	10000	1	13

7.4.2.12. Zoo Veri Kümesi Deney Tasarımı

Taguchi deney tasarımının analizi ile elde edilen ana etkiler grafiği Şekil 7.23.'de gösterilmektedir. Şekildeki ana etki grafiklerine göre, gizli katman sayısı, sıklık faktörü, momentum katsayısı, öğrenme katsayısı, ÇKA iterasyon sayısı ve işlem elemanı sayısı faktörlerinin etkinlik ölçütü üzerinde etkisi olduğu yani bu faktörlerin düzeylerinin değişmesinin test doğruluğunu değiştireceği söylenebilir. Diğer parametrelerin ise düzeylerinin test doğruluğuna herhangi bir etkisi yoktur.



Şekil 7.23. Zoo veri kümesi ana etkiler grafiği.

Zoo veri kümesi için Taguchi deney tasarımı sonucunda elde edilen, faktörlerin etkinlik ölçütü üzerindeki etkilerine göre sıralamaları, ortalamalar ile birlikte Tablo 7.49'da verilmektedir.

Tablo 7.49. Zoo veri kümesi için Taguchi faktör sıralaması.

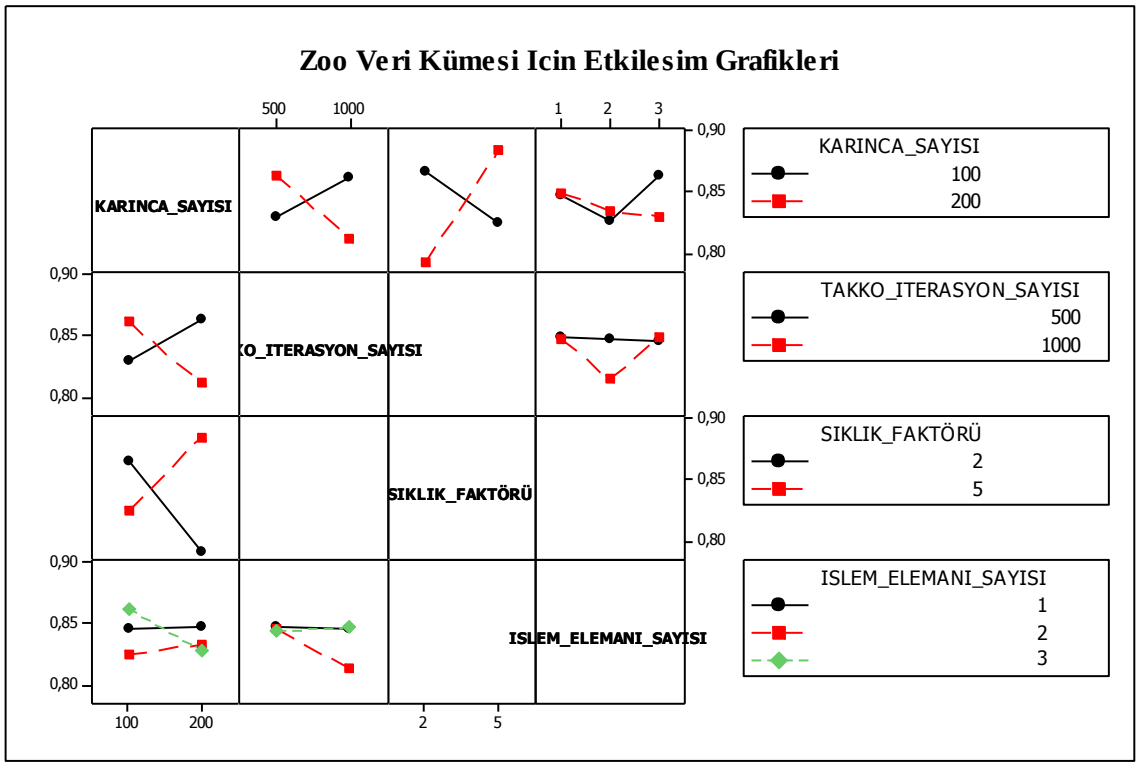
Faktörler Düzeyleri	M	T	f	Q	ρ	α	λ	Epoch	GK	İES
1	0,8453	0,8467	0,8283	0,8429	0,8422	0,8494	0,8525	0,8345	0,9218	0,8478
2	0,8374	0,8361	0,8544	0,8399	0,8405	0,8334	0,8303	0,8483	0,761	0,8299
3										0,8464
Fark	0,0079	0,0105	0,0261	0,003	0,0017	0,0159	0,0222	0,0138	0,1607	0,0179
Sıralama	8	7	2	9	10	5	3	6	1	4

Tablo 7.49 ve Şekil 7.23.'den elde edilen bilgilere göre, bu veri kümesinde faktörlerin test doğruluğu üzerindeki etkilerine göre sıralaması ve en iyi sonuç veren düzeyleri Tablo 7.50.'de gösterilmektedir.

Tablo 7.50. Zoo veri kümesi için faktör sıralamaları ve düzeyleri.

Faktör	GK	f	λ	İES	α	Epoch
Düzey	1	5	0,2	29	0,7	20000

Şekil 7.24. ikili etkileşim grafiklerini göstermektedir. Karınca sayısı-TAKKO iterasyon sayısı, karınca sayısı-sıklık faktörü ve karınca sayısı-gizli katman(lar)daki işlem elemanı sayısı etkileşim eğrileri birbirini kesmektedir. Dolayısıyla bu faktörler arasında ikili etkileşim söz konusudur.



Şekil 7.24. Zoo veri kümesi etkileşim grafiği.

Sonuç olarak, ana etkiler ile birlikte ikili etkileşimler dikkate alındığında; ana etkilere göre ikinci düzeyinde daha etkin olan sıklık faktörü "f", ilk düzeyinde çalıştırırsa test doğruluğu artacaktır. İkili etkileşimler bu faktörün ana etkiler ile belirlenen düzeyinde değişikliğe neden olmuştur.

Ana etkiler sonucu etkisiz çıkan karınca sayısı ve TAKKO iterasyon sayısı faktörleri ise ikili etkileşimleri çıkması nedeni ile etkili hale gelmiştir. İkili etkileşim eğrilerinden faydalanılarak karınca sayısı faktörünün ilk, TAKKO iterasyon sayısı faktörünün ise ikinci düzeyinde çalıştırılması gerektiği sonucuna varılabilir. Feremon sabiti ve buharlaşma katsayısı parametrelerinin ise hangi düzeyde çalıştırıldıklarının test doğruluğu üzerinde etkisi yoktur. Ana etkilerde paralel birer doğruya sahiptirler ve diğer faktörler ile aralarında etkileşim çıkmamıştır.

Geliştirilen algoritmada Zoo veri kümesi için kullanılan düzeyleri tablo 7.51.'de verilmektedir.

Tablo 7.51. Zoo veri kümesi için belirlenen faktör düzeyleri

Faktör	M	T	f	Q	ρ	α	λ	Epoch	GK	İES
Değer	100	1000	2	5	0,9	0,7	0,2	20000	1	29

7.5. Geliştirilen Algoritmanın Test Problemlerine Uygulanması

Geliştirilen algoritma için deneysel tasarım yapılarak her bir referans veri kümesi için faktör düzeyleri belirlenerek önceki bölümde verilmiştir. Elde edilen bu en iyi parametre düzeyleri ile KKO tabanlı kural çıkarım algoritmasına 10-katlı çapraz doğrulama testi uygulanmıştır. 10 çalıştırma sonucunda elde edilen test doğruluklarının ortalamaları, minimum, maksimum değerleri ve standart sapmaları hesaplanmıştır. Kural sayıları ise 10 çalıştırmanın ortalaması olarak ortalama kural sayıları şeklinde ele alınmıştır. Tablo 7.52. ele alınan 12 veri kümesi üzerinde, geliştirilen algoritmanın tahminleyici doğruluk ve kural sayılarını göstermektedir.

Tablo 7.52. TAKKOYSA-sınıflandırıcısı tahminleyici doğruluk ve kural sayıları.

Veri kümesi	Min (%)	Ortalama (%)	Maks (%)	Standart sapma	Ortalama kural sayısı
CRX	92,75	97,54	100	2,17	49,7
Heart-C	80	90,41	100	6,32	45,6
Iris	93,33	99,33	100	2,11	4,8
LBC	89,29	96,84	100	4,23	29,3
Monks-2	97,45	99,07	100	0,78	15,2
Nursery	99,92	99,98	100	0,03	39,7
Pima	97,4	99,61	100	0,88	31,5
Spect-Heart	90,37	94,76	98,4	2,52	17,2
Tic-Tac-Toe	100	100	100	0	16,1
Vote	95,35	98,62	100	1,62	18,8
WBC	97,14	99	100	0,97	24,1
Zoo	94,12	95,3	100	2,06	8,9

Tablo 7.52.'deki sonuçlardan görüldüğü gibi, geliştirilen algoritmada varsayılan (default) sınıf ele alınmadığı halde, algoritma doğru ve anlaşılır kural kümeleri üretebilmektedir. Örnek olması açısından bu bölümde sadece 3 veri kümesinde elde edilen kural kümeleri Tablo 7.53.-7.55.'de verilmektedir. Diğer veri kümeleri için elde edilen kurallar ise EK-1'de yer almaktadır.

Tablo 7.53. Iris veri kümesinde üretilen en iyi kural kümesi.

Kural No	**Kalite**	**Uygunluk**	**Elde edilen örnek kural kümesi**
1	0,9548	1	Eğer (sepal length≤6,15) ve (petal length≤2,45) ise sınıf "**setosa**"
2	0,9646	0,9342	Eğer (0,18<petal width ≤1,75) ise sınıf "**versicolour**"
3	0,9151	0,9236	Eğer (petal length>4.75) ise sınıf "**virginica**"
4	0,9613	0,5844	Eğer (sepal length≤5,55 veya sepal length>6,15) ve (petal length>2,45) ise sınıf "**virginica**".

Eğitim Doğruluğu:	99,26	Güvenilirlik:	100
Test Doğruluğu:	100	Destek:	100
ÇKA En İyi MSE değeri:	0,05	Genellik:	100

Tablo 7.54. Tic-Tac-Toe veri kümesinde üretilen en iyi kural kümesi.

Kural No	Kalite	Uygunluk	Elde edilen örnek kural kümesi
1	0,9999	0,4345	Eğer (middle middle square=0 veya b) ise sınıf "**pozitif**"
2	0,9999	0,3269	Eğer (top telf square=x veya b) ve (top right square=x veya b) ise sınıf "**pozitif**"
3	1	0,3085	Eğer (bottom right square=x) ise sınıf "**negatif**"
4	0,9999	0,3069	Eğer (bottom left square=o) ise sınıf "**negatif**".
5	9,9999	0,2976	Eğer (bottom left square=x) ise sınıf "**pozitif**"
6	0,9999	0,2899	Eğer (top left square=x veya o) ve (top rihgt square=x veya b) ise sınıf "**pozitif**"
7	0,9999	0,2848	Eğer (bottom right square=x veya o) ise sınıf "**negatif**"
8	0,9999	0,2798	Eğer (top middle square=x veya o) ve (middle left square=x veya o) ise sınıf "**negatif**".
9	0,9999	0,2712	Eğer (middle left square=o veya b) ve (middle middle square=x veya o) ise sınıf "**negatif**"
10	0,9999	0,2542	Eğer (top left square=x veya o) ve (middle left square=o veya b) ve (bottom left square=x veya o) ise sınıf "**negatif**"
11	0,9999	0,2377	Eğer (top middle square=o veya b) ve (top right square=x veya o) ve (middle left square=x veya b) ise sınıf "**negatif**"

Eğitim Doğruluğu:	100	Güvenilirlik:	100
Test Doğruluğu:	100	Destek:	100
ÇKA En İyi MSE değeri:	0,000002	Genellik:	100

Tablo 7.55. Zoo veri kümesinde üretilen en iyi kural kümesi.

Kural No	Kalite	Uygunluk	Elde edilen örnek kural kümesi
1	0,9258	1	Eğer (feathers=1) ve (egss=1) ve (backbone=1) ve (legs=2 veya 4 veya 8) ve (tail=1) ise sınıf "**2**"
2	0,9327	1	Eğer (eggs=1) ve (aquatic=1) ve (breathes=0) ve (legs=0 veya 2 veya 6 veya 8) ve (tail=1) ise sınıf "**4**"
3	0,9415	1	Eğer (milk=0) ve (aquatic=1) ve (toothed=1) ve (bacbone=1) ve (fins=0) ve (legs=4 veya 8) ve (catsize=0) ise sınıf "**5**"
4	0,9569	1	Eğer (eggs=1) ve (milk=0) ve (bacbone=0) ve (breathes=1) ve (legs=5 veya 6) ise sınıf "**6**".
5	0,9333	0,9844	Eğer (feathers=0) ve (milk=0) ve (bacbone=1) ve (fins=0) ve (legs=0 veya 4 veya 5 veya 8) ve (tail=1) ve (domestic=0) ise sınıf "**3**"
6	0,9334	0,9259	Eğer (hair=1) ve (milk=1) ve (airbone=0) ve (bacbone=1) ve (legs=2 veya 4 veya 5) ise sınıf "**1**"
7	0,9876	0,9259	Eğer (hair=1) ve (milk=1) ve (toothed=1) ve (bacbone=1) ve (breathes=1) ve (venomous=0) ve (legs=2 veya 4 veya 5 veya 6) ise sınıf "**1**"
8	0,9742	0,8519	Eğer (eggs=0) ve (milk=1) ve (toothed=1) ve (legs=0 veya 4 veya 6 veya 8) ise sınıf "**1**".
9	0,9522	0,6667	Eğer (hair=0) ve (milk=0) ve (airbone=0) ve (aquatic=1) ve (preator=1) ve (backbone=0) ve (breathes=0) ve (fins=0) ve (legs=4 veya 5 veya 6 veya 8) ve (tail=0) ve (domestic=0) ise sınıf "**7**"
10	0,9922	0,6667	Eğer (fethers=0) ve (milk=0) ve (toothed=0) ve (bacbone=0) ve (fins=0) ve (legs=0 veya 5 veya 8) ve (domestic=0) ise sınıf "**7**"

Eğitim Doğruluğu:	100	Güvenilirlik:	100
Test Doğruluğu:	100	Destek:	100
ÇKA En İyi MSE değeri:	0,00001	Genellik:	100

7.6. Performans Karşılaştırmaları

Geliştirilen TAKKOYSA-sınıflandırıcısın performansı iki aşamada değerlendirilmiştir. İlk aşamada, geliştirilen yaklaşım Weka 3.5 yazılımının dört popüler kural tabanlı makine-öğrenme algoritması ile karşılaştırılmıştır. Bu algoritmalar "C4.5" [41], "PART" [183], "DecisionTable" [184] ve "NBTree"dir [185]. C4.5 algoritması, uygulamada yaygın olarak kullanılan bir karar ağacı algoritmasıdır [66]. PART, sınıflandırma kuralları üretmekte yetenekli bir kural-öğrenme yapısıdır. DecisionTable algoritması, basit bir karar tablosu oluşturmak ve kullanmakta bir sınıflandırıcıdır. NBTree algoritması, birçok makine-öğrenme araştırmacısı tarafından çalışılmış bir algoritmadır ve Bayes tekniklerini kullanır.

Geliştirilen TAKKOYSA-sınıflandırıcısı ile karşılaştırılan klasik makine öğrenme algoritmalarının bulduğu sonuçlar arasındaki farkın istatistiksel olarak anlamlı olup olmadığını göstermek için hipotez testlerinden faydalanmıştır. Hipotez testleri, araştırma sonucunda elde edilen değerlerin istatistiksel olarak önem taşıyıp taşımadığını, başka bir ifade ile anlamlı olup olmadığını test etmek için kullanılan yöntemlerdir.

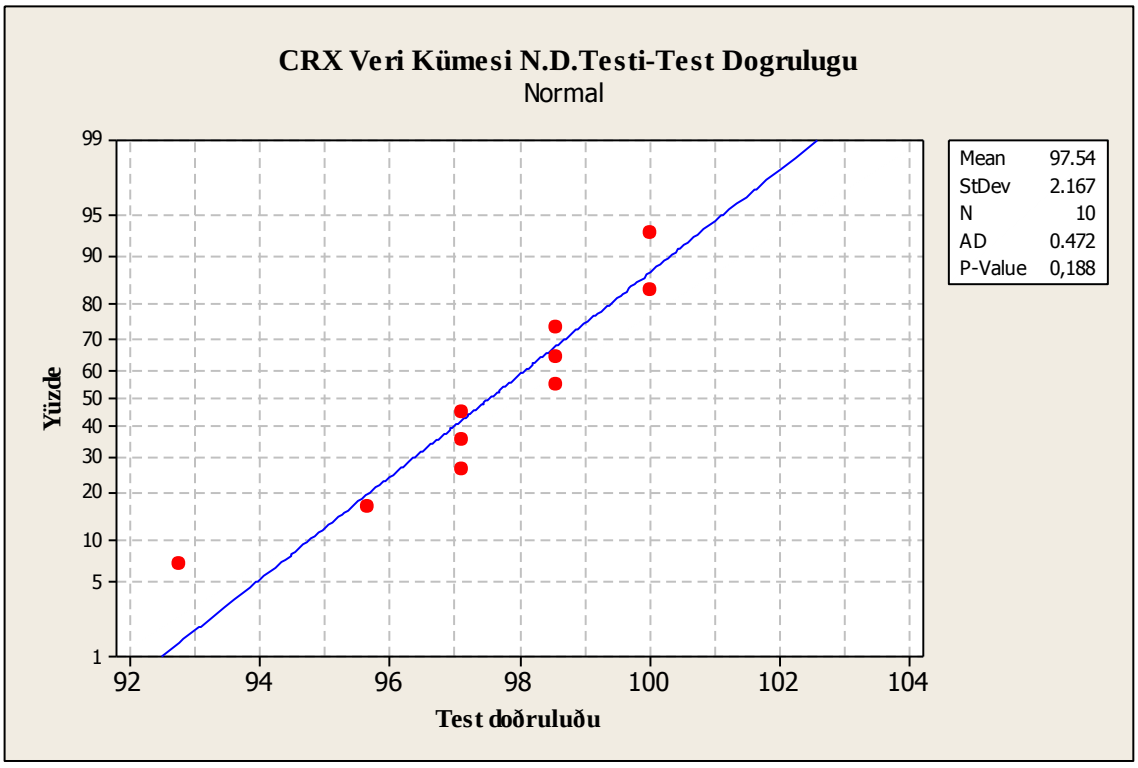
Gerçekleştirilen hipotez testlerinde önemlilik seviyesi (α) dikkate alınarak hesaplanan *p* değerlerine göre α=0,05 önemlilik düzeyinde geliştirilen algoritmanın elde ettiği sonuçların klasik makine öğrenme algoritmalarından farklı olup olmadığı araştırılmıştır.

Hipotez testleri temelde parametrik ve parametrik olmayan olmak üzere iki gruba ayrılmaktadır. Parametrik testler ortalama, varyans gibi ölçütler üzerinde çalışırken; parametrik olmayan testler verilerin numara sıraları üzerinden işlem yapmaktadır. Parametrik testler her ne kadar parametrik olmayan testlere göre daha güçlü testler olsalar da kullanılabilmeleri için bazı varsayımları sağlamaları gerekir.

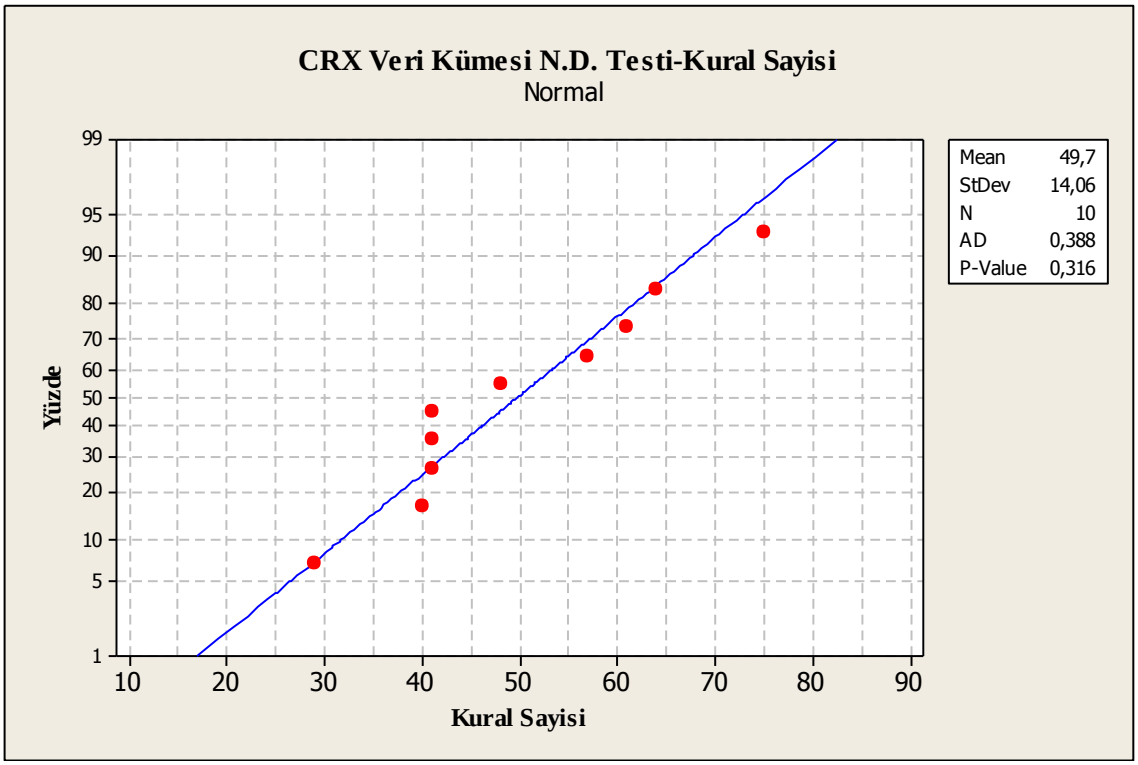
Bu nedenle uygun hipotez testinin seçimi verilerin elde edilmesi ile ilgili durumlara göre yapılır. İlk bakılması gereken durum, örneklem büyüklüğüdür, çünkü örnek sayısı arttıkça kullanılan testin gücü ve güvenilirliği de artar. Genellikle veri sayısı 30'un altında olduğunda parametrik olmayan testler tercih edilir. Veri sayısının 30'un üzerinde olması halinde ise diğer varsayımlar da muhakkak değerlendirilmelidir.

Geliştirilen algoritmada deney tasarımıyla elde edilen parametre düzeyleriyle algoritmalar 10-katlı çapraz geçerlilik testi gereği 10 kez çalıştırılmıştır. Dolayısıyla her veri kümesi için 10 örnek söz konusudur. Örnek sayısının 30'dan az olması nedeniyle parametrik olmayan test kullanmak genel olarak önerilse de diğer durumları da incelemekte fayda vardır.

Geliştirilen algoritmada elde edilen test doğrulukları ve kural sayıları ***sürekli nicel*** verilerdir. Algoritma, kendisi gibi sınıflandırıcılarla karşılaştırıldığı için karşılaştırmaya tabi olan algoritmalar ***bağımlı***dırlar. Ancak bu çıkarımlar kullanılacak teste karar verilmesinde yeterli olmayıp, verilerin dağılımına da bakmak gerekir. Çünkü parametrik testlerin uygulanabilmesi için dağılımın normal veya normale yakın olması gerekir [186]. Bu nedenle algoritma ile elde edilen test doğrulukları ve kural sayılarının normal dağılıma uyup uymadıkları test edilmiştir. Şekil 7.25 ve 7.26'da örnek olarak CRX veri kümesinde sırasıyla test doğruluğu ve kural sayısı etkinlik ölçütleri için normal olasılık testi sonuçları gösterilmektedir.



Şekil 7.25. CRX veri kümesi test doğruluğu normal dağılım testi.



Şekil 7.26. CRX veri kümesi kural sayısı normal dağılım testi.

Şekillerden de görülebileceği gibi bazı noktalar normal olasılık eğrisinin dışında çıkmıştır. Ayrıca, her iki testin de p değerleri α=0,05 önemlilik seviyesinin üzerinde çıkmıştır. Bu nedenle test doğruluklarının ve kural sayılarının normal dağılıma uymadığı sonucuna varılabilir. Diğer veri kümeleri için de normal olasılık testi gerçekleştirilmiş ve bir iki veri kümesi dışında sonuçların normal olasılık dağılımına uymadığı gözlemlenmiştir (EK-2).

Örnek sayısı 30'dan küçük olduğu için ve çoğu veri kümesinde veriler normal dağılıma uymadığı için, tez çalışmasında hipotez testi olarak, normal dağılıma uymayan, sürekli nicel, bağımlı veriler için uygun olan ve ikili karşılaştırmalarda kullanılan [186] "Mann Whitney U" testi kullanılmıştır. Bu test, iki grup medyanları arasındaki farkın anlamlı olup olmadığı hakkında çıkarsama yapmakta kullanılmaktadır. Medyan bir veri dizisinin küçükten büyüğe veya büyükten küçüğe sıralanmasından sonra bu dizinin tam ortasında bulunan değerdir. Eğer veri büyüklüğü tek sayılı ise medyan verilen bir değere, yani ortadaki değere eşit olur. Eğer veri büyüklüğü çift sayılı ise, medyan orta iki değerin ortalaması alınarak bulunur.

Değerlendirmenin ikinci aşamasında, kural çıkarım algoritmasının değerlendirmeye alınan 12 veri kümesindeki doğruluk ve ortalama kural sayıları, günümüz literatüründe mevcut olan çeşitli kural-tabanlı sınıflandırıcılarla karşılaştırılmıştır. Literatürde yer alan bu algoritmaların on-katlı çapraz geçerlilik testinin her bir katında elde edilen sonuçlar mevcut olmadığından TAKKOYSA-sınıflandırıcısı ile literatür-sınıflandırıcılarının buldukları sonuçların istatistiksel olarak aralarında anlamlı bir farklılık olup olmadığını araştırmak mümkün olmamıştır. Ancak elde edilen tahminleyici doğruluk ve ortalama kural sayıları, sayısal olarak değerlendirilmiştir.

Tez çalışmasında 12 veri kümesinin analizler kapsamına alınması ve karşılaştırılacak algoritmaların çok olması nedeniyle, karşılaştırma sonuçlarını tek bir tabloda göstermek mümkün değildir. Bu nedenle karşılaştırmalar her veri kümesi için ayrı olarak ele alınmıştır. İlerleyen bölümlerde her bir veri kümesi için karşılaştırma sonuçları verilmektedir.

7.6.1. CRX Veri Kümesi İçin Karşılaştırma Sonuçları

Tablo 7.56. geliştirilen algoritmaya 10-katlı çapraz geçerlilik testi uygulanması sonucu elde edilen sonuçların NBTree, PART, DecisionTable (DT) ve C4.5 algoritmaları ile karşılaştırılmasına ilişkin sonuçları vermektedir. Karşılaştırmalar minimum, ortalama ve maksimum test doğruluğu, test doğruluğu standart sapması ve ortalama kural sayıları dikkate alınarak gerçekleştirilmiştir.

Tablo 7.56. CRX veri kümesi klasik sınıflandırıcılar ile karşılaştırma sonuçları.

Veri kümesi		NBTree	DT	PART	C4.5	Geliştirilen Algoritma
	Minimum	73,913	78,261	76,812	79,71	**92,75**
	Ortalama	84,348	85,072	85,362	86,087	**97,54**
CRX	Maksimum	92,754	91,304	94,203	91,304	**100**
	Standart sapma	5,584	4,736	4,804	3,754	**2,17**
	Ort. kural sayısı	**7,8**	40,9	32,2	18,2	49,7

Tablodan da görüldüğü gibi tahminleyici doğruluklar dikkate alındığında geliştirilen algoritma büyük bir farkla minimum, ortalama ve maksimum değerlerin üçünde de

karşılaştırmaya tabi tutulan dört algoritmadan daha iyi sonuçlar bulmuştur. On katlı çapraz doğrulamada sonuçlar arasındaki en düşük standart sapma da geliştirilen algoritmaya aittir. Bu da ele alınan eğitim ve test kümesi değişse de kural çıkarım algoritmasının kaliteli çözümler üretebildiğini göstermektedir.

Çıkarılan ortalama kural sayısı dikkate alındığında ise, geliştirilen algoritma diğer algoritmalardan daha fazla kural içermektedir. En az kural, 7,8 ile NBTree algoritmasına aittir, ancak bu algoritmanın tahminleyici doğruluğu ile geliştirilen algoritmanın elde ettiği doğruluk arasında büyük bir fark vardır. Elde edilen kural sayısı ile DecisionTable algoritmasının kural sayısı arasında önemli bir fark görülmemektedir.

Elde edilen sonuçların daha iyi değerlendirilebilmesi ve aralarında anlamlı bir farklılık olup olmadığının belirlinebilmesi için gerçekleştirilen Mann Whitney U testi sonuçları Tablo 7.57.'de verilmiştir. Bu test ile geliştirilen algoritma ve her bir klasik makine öğrenme algoritması medyan değerleri arasındaki farkın anlamlı olup olmadığı test edilmiştir. Tabloda algoritmaların medyan değerleri ve p değerleri gösterilmektedir.

Tablo 7.57. CRX veri kümesi için Mann Whitney U testi sonuçları.

	Etkinlik Ölçütü			
	Test Doğruluğu		Kural Sayısı	
Algoritma	Medyan	P değeri	Medyan	P değeri
TAKKOYSA-sınıflandırıcısı NBTree	97,825 85,507	0,0002	44,5 5	(-)0,0002
TAKKOYSA-sınıflandırıcısı Decision Table	97,825 85,407	0,0002	44,5 36,5	**0,089**
TAKKOYSA-sınıflandırıcısı PART	97,825 85,508	0,0002	44,5 33,5	(-)0,0022
TAKKOYSA-sınıflandırıcısı C4.5	97,825 86,957	0,0002	44,5 18	(-)0,0002

Tablo 7.57.'deki test doğruluğu-medyan değerleri incelendiğinde geliştirilen algoritmanın medyan değerinin diğer dört algoritmanın medyan değerinden de büyük olduğu görülür. Ancak medyanlar arasındaki farkın istatistiksel olarak anlamlı olup

olmadığını söylemek için p değerlerine bakmak gerekir. Eğer p-değeri α=0,05 önemlilik değerinden küçükse, karşılaştırılan algoritmaların medyan değerleri arasında istatistiksel olarak anlamlı bir farklılık var demektir. Test doğruluğu p değerleri incelendiğinde geliştirilen algoritmanın elde ettiği tahminleyici doğruluklar arasında istatistiksel olarak anlamlı bir farklılık görülmektedir ($p<0,05$). Geliştirilen algoritmanın test doğruluğu medyan değerinin diğer algoritmalardan daha büyük olması geliştirilen algoritmanın istatistiksel olarak daha iyi sonuçlar üretebildiğini göstermektedir.

Tablo 7.57.'de (-) işaretli değerler, karşılaştırılan algoritmanın, geliştirilen kural çıkarım algoritmasından daha iyi sonuç verdiğini ifade etmektedir. Çıkarılan kural sayıları dikkate alındığında, karşılaştırmada kullanılan dört algoritmanın da medyan değerlerinin geliştirilen algoritmadan küçük olduğu görülür. p değerleri incelendiğinde DecisionTable algoritması dışında kalan üç algoritmanın istatistiksel olarak geliştirilen algoritmadan daha az sayıda kural ürettiği söylenebilir ($p<0,05$).

Tablodaki koyu işaretlenmiş değerler, ilgili veri kümesinde, karşılaştırılan algoritmaların medyan sonuçlarının birbirinden farklı olduğunu ispatlamanın istatistiksel olarak mümkün olmadığını göstermektedir, çünkü bu değerler 0,05 önemlilik düzeyinden büyüktür. Kural sayısı etkinlik ölçütünde DecisionTable algoritması ile geliştirilen algoritmanın elde ettiği kural sayıları arasında istatistiksel olarak bir anlamlılık yoktur, çünkü p değeri önemlilik düzeyinden büyüktür ($p>0,005$).

Tablo 7.58. geliştirilen TAKKOYSA-sınıflandırıcısı ile günümüz literatüründe yer alan kural-tabanlı sınıflandırıcıların elde ettikleri tahminleyici doğruluklar ve ortalama kural sayılarının karşılaştırılmasını vermektedir. Bazı algoritmaların ismi olmadığından bu algoritmalar için tabloda sadece referans yer almaktadır.

Aynı şekilde, kural sayılarında bazı algoritmalara ait bilgi yer almamaktadır. Bunun nedeni ilgili algoritmanın ya her sınıf için sadece bir kural çıkarıyor olması veya kural sayısı bilgisinin bulunmamasıdır. *** ile gösterilen sınıflandırıcılar ise yapay sinir ağlarından kural çıkarım algoritmalarını ifade etmektedir.

Tablo 7.58. CRX veri kümesinde TAKKOYSA-sınıflandırıcısının kural-tabanlı sınıflandırıcılar ile karşılaştırma sonuçları.

Veri Kümesi	**Algoritma**	**Referans**	**Kural Sayısı**	**Ortalama Doğruluk**
CRX	DOEA	Tan et al. (2006a)	5	85,48
	CORE	Tan et al. (2006b)	4,75	86,49
	-	Carvalho ve Freitas (2002)	-	86,12
	DCC	Tan et al. (2005)	4	86,28
	C4.5/GA-small	Carvalho ve Freitas (2004)	-	90,89
	C4.5/GA-large-SN	Carvalho ve Freitas (2004)	-	91,66
	MEPAR-miner	Baykasoğlu ve Özbakır (2007)	-	96,96
	MOGGP	Pappa ve Freitas (2008)	-	83,33
	PSO/ACO2	Holden ve Freitas (2007)	-	86,19
	cAnt-Miner	Otero et al., (2008)	-	85,56
	cAnt-Miner-MDL	Otero et al., (2009)	-	85,11
	cAnt-Miner 2	Otero et al., (2009)	-	85,79
	cAnt-Miner 2-MDL	Otero et al., (2009)	-	85,9
	GARC	Chen et al. (2006)	21	82,5
	-	Pitangui ve Zaverucha (2006)	**1,4**	92,41
	***CGA	Hruschka ve Ebecken (2006)	2	76,26
	***G-REX oracle	Johansson vd., 2006	-	72,7
	TAKKOYSA-sınıflandırıcısı		49,7	**97,54**

Tablo 7.58. incelendiğinde ortalama doğruluk etkinlik ölçütünde en iyi sonucu TAKKOYSA-sınıflandırıcısının verdiği görülmektedir. Ancak çıkarılan kural sayısı diğer algoritmalara göre daha fazladır. Bunun nedeni geliştirilen algoritmanın ikili diziler üzerinde çalışmasıdır.

TAKKOYSA-sınıflandırıcısı, diğer yapay sinir ağlarından kural çıkaran algoritmalar ile karşılaştırıldığında ise tahminleyici doğruluklar arasında büyük bir fark görülmektedir. Diğer algoritmalar % 70'lerde doğruluk elde ederken, geliştirilen algoritma yaklaşık % 98 doğruluk elde etmiştir. Geliştirilen algoritmanın kural sayısı diğer iki algoritmadan yüksek çıkmıştır. Kural sayısının fazla çıkmasının, algoritmanın ikili diziler üzerinde

çalışmasından başka diğer bir nedeni de, algoritmada ilk amaç olarak tahminleyici doğruluğun maksimizasyonunun ele alınmasıdır. Algoritmada ele alınan amaç değiştirilerek veya çoklu amaçlar algoritmaya eklenerek kural sayıları düşürülebilir.

7.6.2. Heart-C Veri Kümesi İçin Karşılaştırma Sonuçları

Geliştirilen algoritmaya 10-katlı çapraz geçerlilik testi uygulanması sonucu elde edilen sonuçların dört farklı makine öğrenme algoritması ile karşılaştırma sonuçları Tablo 7.59.'da gösterilmektedir. Her bir algoritma ile elde edilen minimum, ortalama ve maksimum test doğruluğu, test doğruluğu standart sapması ve ortalama kural sayıları tabloda yer almaktadır.

Tablo 7.59. Heart-C veri kümesi klasik sınıflandırıcılar ile karşılaştırma sonuçları.

Veri kümesi		NBTree	DT	PART	C4.5	Geliştirilen Algoritma
	Minimum	35,484	50	38,71	41,935	**80**
	Ortalama	53,183	54,817	55,505	52,516	**90,41**
Heart-C	Maksimum	63,333	63,333	73,333	63,333	**100**
	Standart sapma	9,09	**4,277**	9,512	6,804	6,32
	Ort. kural sayısı	**7,9**	10,1	40,3	38,7	45,6

TAKKOYSA-sınıflandırıcısının klasik makine öğrenme algoritmalarıyla karşılaştırma sonuçlarından görüldüğü gibi, algoritma minimum, ortalama ve maksimum tahminleyici doğruluklarda neredeyse diğer algoritmalardan iki kat daha iyi sonuçlar elde etmiştir. Standart sapmalara göre 10-katlı çapraz doğrulamada, birbirine yakın sonuçları DecisionTable algoritması elde etmiştir. Ancak bulduğu sonuçlar % 50-63 arasında kalitesiz sonuçlardır ve test veri kümesinin ancak yarısını ifade edebilmektedir. Geliştirilen algoritma standart sapmalar dikkate alındığında ikinci sırada yer almaktadır. Dolayısıyla algoritmanın iyi çözümler üretmekte başarılı olduğu söylenebilir.

Çıkarılan ortalama kural sayısı dikkate alındığında ise TAKKOYSA-sınıflandırıcısı, diğer algoritmalardan daha fazla kural içermektedir. En az kural, 7,9 ile NBTree algoritmasına aittir, ancak bu algoritmanın tahminleyici doğruluğu % 53'lerde iken geliştirilen algoritma ortalama % 90'ın üzerinde doğrulukta kurallar üretmektedir. Elde

edilen kural sayısı ile PART ve C4.5 algoritmaları ile elde edilen kural sayıları arasında büyük bir fark görülmemektedir. Ancak bunu istatistiksel olarak değerlendirmek gerekir.

Tablo 7.60. elde edilen sonuçların daha iyi değerlendirilebilmesi ve aralarında anlamlı bir farklılık olup olmadığının belirlenebilmesi için gerçekleştirilen Mann Whitney U testi sonucu elde edilen medyan ve p değerlerini göstermektedir.

Tablo 7.60. Heart-C veri kümesi için Mann Whitney U testi sonuçları.

	Etkinlik Ölçütü			
	Test Doğruluğu		Kural Sayısı	
Algoritma	Medyan	P değeri	Medyan	P değeri
TAKKOYSA-sınıflandırıcısı NBTree	91,67 56,67	0,0002	43 8	(-)0,0002
TAKKOYSA-sınıflandırıcısı Decision Table	91,67 53.33	0,0002	43 9	(-)0,0002
TAKKOYSA-sınıflandırıcısı PART	91,67 56,67	0,0002	43 40	**0,2265**
TAKKOYSA-sınıflandırıcısı C4.5	91,67 53,33	0,0002	43 39	(-)0,0413

Tabloda verilen test doğruluğu-medyan değerleri incelendiğinde geliştirilen algoritmanın medyan değerinin diğer dört makine öğrenme algoritmasının medyan değerinden daha büyük olduğu görülmektedir. Ancak medyanlar arasındaki farkın istatistiksel olarak anlamlı olup olmadığını söylemek için p değerlerine bakmak gerekir. Test doğruluğu p değerleri incelendiğinde geliştirilen algoritmanın elde ettiği tahminleyici doğruluklar ile diğer algoritmaların doğrulukları arasında istatistiksel olarak anlamlı bir farklılık görülmektedir ($p<0,05$). Algoritmanın medyan değerinin de diğer algoritmalardan daha büyük olması algoritmanın istatistiksel olarak daha iyi sonuçlar üretebildiğini göstermektedir.

Çıkarılan kural sayıları dikkate alındığında karşılaştırmada kullanılan dört algoritmanın da kural sayısı medyan değerlerinin TAKKOYSA-sınıflandırıcısından küçük olduğu görülmektedir. p değerleri incelendiğinde PART algoritması dışında kalan üç

algoritmanın istatistiksel olarak geliştirilen algoritmadan daha az sayıda kural ürettiği söylenebilir ($p<0,05$). Kural sayısı etkinlik ölçütünde PART algoritması ile geliştirilen algoritmanın elde ettiği kural sayıları arasında istatistiksel olarak bir anlamlılık yoktur, çünkü p değeri önemlilik düzeyi olan 0,05'den büyüktür.

Tablo 7.61. TAKKOYSA-sınıflandırıcısı ile günümüz literatüründe yer alan kural-tabanlı sınıflandırıcıların karşılaştırma sonuçlarını vermektedir. Ortalama kural sayıları ve ortalama test doğruluğu karşılaştırmada kullanılan ölçütlerdir.

Bazı algoritmalarda kural sayıları ya değerlendirilmemiş, ya da her sınıf için bir kural çıkarılmıştır. Bu nedenle bu algoritmaların kural sayısı sütunları boş bırakılmıştır.
Tablo 7.61.'de yer alan ortalama doğruluk sütunu incelendiğinde en iyi sonucu geliştirilen yapay sinir ağlarından kural çıkarım algoritmasının verdiği görülmektedir. Kural kümesinde en az kuralı 2,6 kural ile Pitangui ve Zavercha (2006) elde etmiştir. En çok kuralı ise 290,4 kural ile CMAR algoritması üretmiştir. Geliştirilen algoritma ise 45,6 kural ile ortalama bir kural kümesi elde etmiştir. En yüksek test doğruluğu ile ortalama bir kural sayısı, algoritmanın başarısının bir ölçüsü olarak kabul edilebilir.

Algoritma, bir diğer yapay sinir ağları üzerinde çalışan G-REX oracle ile karşılaştırıldığında ise tahminleyici doğruluklar arasında büyük bir fark görülmektedir. G-REX oracle algoritması % 81 doğruluk elde ederken, TAKKOYSA-sınıflandırıcısı % 90,41 doğruluk elde etmiştir.

Tablo 7.61. Heart-C veri kümesinde TAKKOYSA-sınıflandırıcısının kural-tabanlı sınıflandırıcılar ile karşılaştırma sonuçları.

Veri Kümesi	**Algoritma**	**Referans**	**Kural Sayısı**	**Ortalama Doğruluk**
Heart-C	DOEA	Tan et al. (2006a)	6	79,84
	CORE	Tan et al. (2006b)	6,24	80,77
	MEPAR-miner	Baykasoğlu ve Özbakır (2007)	-	87,78
	-	Wang ve Zhang (2005)	-	58,85
	DCC	Tan et al. (2005)	8	80,01
	MOGGP	Pappa ve Freitas (2008)	-	76,46
	Ant-Miner	Parpinelli et al. (2002b)	9,5	59,67
	ACO-Miner	Wang ve Feng (2004)	7,29	78,28
	Unordered Rule set Ant-Miner	Smaldon ve Freitas (2006)	11	56,38
	E-Ant-Miner	Ji et al. (2006)	8,5	56,9
	-	Nalini ve Balasubramanie (2008)	7,2	81,8
	PSO/ACO2	Holden ve Freitas (2007)	-	80,53
	GARC	Chen et al. (2006)	23	80,2
	RMR	Thabtah ve Cowling (2007)	75	84,74
	TFPC	Coenen ve Leng (2007)	98,8	51,4
	CMAR	Coenen ve Leng (2007)	290,4	54,4
	CBA	Coenen ve Leng (2007)	52,9	57,3
	-	Pitangui ve Zaverucha (2006)	**2,6**	86,88
	***G-REX oracle	Johansson vd., 2006	-	81
	TAKKOYSA-sınıflandırıcısı		45,6	**90,41**

7.6.3. Iris Veri Kümesi İçin Karşılaştırma Sonuçları

Tablo 7.62. TAKKOYSA-sınıflandırıcısına 10-katlı çapraz geçerlilik testi uygulanması sonucu elde edilen sonuçların klasik makine öğrenme algoritmalarından NBTree, PART, DecisionTable ve C4.5 algoritmaları ile karşılaştırılması sonuçlarını vermektedir. Karşılaştırmalar minimum, ortalama ve maksimum test doğruluğu, test

doğruluğu standart sapması ve ortalama kural sayıları dikkate alınarak gerçekleştirilmiştir.

Tablo 7.62. Iris veri kümesi klasik sınıflandırıcılar ile karşılaştırma sonuçları.

Veri kümesi		NBTree	DT	PART	C4.5	Geliştirilen Algoritma
	Minimum	86,67	86,67	86,67	86,67	**93,33**
	Ortalama	94	92,67	94	96	**99,33**
Iris	Maksimum	**100**	**100**	**100**	**100**	**100**
	Standart sapma	4,92	5,84	5,84	5,62	**2,11**
	Ort. kural sayısı	**2,9**	4,5	3,8	4,7	4,8

Tablodan da görüldüğü gibi geliştirilen kural çıkarım algoritması tahminleyici doğruluklar dikkate alındığında diğer algoritmalardan daha iyi sonuçlar elde etmiştir. 10-katlı çapraz doğrulama sonucu elde edilen kuralların doğrulukları diğer algoritmalara göre standart sapması düşük olması nedeniyle birbirleriyle daha tutarlıdır. Bu bakımdan geliştirilen algoritma sağlamdır denilebilir.

Çıkarılan ortalama kural sayısı dikkate alındığında ise TAKKOYSA-sınıflandırıcısı dahil beş algoritmanın birbirine yakın sayıda kural içeren kural kümeleri ürettikleri söylenebilir. Ancak aradaki küçük farkın istatistiksel olarak anlamlı olup olmadığını test etmek gerekir.

Elde edilen sonuçların daha iyi değerlendirilebilmesi ve aralarında anlamlı bir farklılık olup olmadığının belirlenebilmesi için gerçekleştirilen Mann Whitney U testi sonuçları Tablo 7.63.'de gösterilmektedir. Bu test, geliştirilen algoritma ile diğer her bir klasik makine öğrenme algoritması medyan değerleri arasındaki farkın anlamlı olup olmadığını test etmektedir. Tabloda algoritmaların medyan değerleri ve p değerleri gösterilmektedir.

Tablo 7.63. Iris veri kümesi için Mann Whitney U testi sonuçları.

	Etkinlik Ölçütü			
	Test Doğruluğu		Kural Sayısı	
Algoritma	Medyan	P değeri	Medyan	P değeri
TAKKOYSA-sınıflandırıcısı	100	0,0126	5	(-)0,0046
NBTree	93,33		3,5	
TAKKOYSA-sınıflandırıcısı	100	0,0126	5	**0,8798**
Decision Table	93,33		4,5	
TAKKOYSA-sınıflandırıcısı	100	0,0376	5	**0,064**
PART	93,33		3	
TAKKOYSA-sınıflandırıcısı	100	**0,2123**	5	**0,8206**
C4.5	100		5	

Tablo 7.63.'de verilen test doğruluğu-medyan değerleri incelendiğinde TAKKOYSA-sınıflandırıcısının medyan değerinin C4.5 algoritması ile aynı, diğer üç algoritmanın medyan değerinden ise büyük olduğu görülmektedir. Test doğruluğu p değerleri incelendiğinde geliştirilen algoritmanın elde ettiği tahminleyici doğruluk ile PART, DecisionTable ve NBTree algoritmalarının elde ettiği doğruluklar arasında istatistiksel olarak anlamlı bir farklılık görülmektedir ($p<0,05$). TAKKOYSA-sınıflandırıcısının medyan değerinin de daha büyük olması geliştirilen algoritmanın diğer algoritmalardan istatistiksel olarak daha iyi sonuçlar üretebildiğini göstermektedir. Iris veri kümesinde C4.5 algoritması ile geliştirilen algoritma arasında tahminleyici doğruluk yönünden istatistiksel olarak bir fark görülmemektedir, çünkü elde edilen p değeri 0,05'den büyüktür.

Çıkarılan kural sayıları dikkate alındığında, C4.5 dışındaki tüm algoritmaların medyan değerlerinin geliştirilen algoritmadan küçük olduğu görülür. p değerleri incelendiğinde ise sadece NBTree algoritmasının geliştirilen algoritmadan daha az sayıda kural ürettiği söylenebilir ($p<0,05$).

Tablodaki koyu işaretlenmiş değerler, ilgili veri kümesinde, karşılaştırılan algoritmaların medyan sonuçlarının birbirinden farklı olduğunu ispatlamanın istatistiksel olarak mümkün olmadığını göstermektedir, çünkü bu değerler 0,05 önemlilik düzeyinden büyüktür. Kural sayısı etkinlik ölçütünde DecisionTable, C4.5 ve

PART algoritmaları ile TAKKOYSA-sınıflandırıcısının elde ettiği kural sayıları arasında istatistiksel olarak bir anlamlılık yoktur, çünkü p değeri önemlilik düzeyinden büyüktür (p>0,005).

Tablo 7.64. TAKKOYSA-sınıflandırıcısı ile günümüz literatüründe yer alan kural tabanlı sınıflandırıcıların elde ettikleri tahminleyici doğruluklar ve ortalama kural sayılarının karşılaştırma sonuçlarını vermektedir. Kural sayılarında bazı algoritmalara ait bilgi yer almamaktadır. Bunun nedeni ilgili algoritmanın ya her sınıf için sadece bir kural çıkarıyor olması veya kural sayısı bilgisinin bulunmamasıdır. *** ile gösterilen sınıflandırıcılar ise yapay sinir ağlarından kural çıkarım algoritmalarını ifade etmektedir.

Tablo 7.64. Iris veri kümesinde TAKKOYSA-sınıflandırıcısının kural-tabanlı sınıflandırıcılar ile karşılaştırma sonuçları.

Veri Kümesi	**Algoritma**	**Referans**	**Kural Sayısı**	**Ortalama Doğruluk**
Iris	DOEA	Tan et al. (2006a)	4	92,81
	CORE	Tan et al. (2006b)	**3,77**	96,61
	DCC	Tan et al. (2005)	4	96,73
	PSO/ACO2	Holden ve Freitas (2007)	-	97,33
	GARC	Chen et al. (2006)	7	94
	RMR	Thabtah ve Cowling (2007)	15	93,87
	TFPC	Coenen ve Leng (2007)	14,8	95,3
	CMAR	Coenen ve Leng (2007)	61,6	93,3
	CBA	Coenen ve Leng (2007)	11,6	94
	AntMiner+	Martens et al. (2007)	4,2	94,51
	***RullExt	Zhang vd. (1996)	3	97,33
	***-	Santos vd.(2000)	5	93,33
	***CGA	Hruschka ve Ebecken (2006)	3	96
	***-	Fu ve Wang (2001)	3	97,33
	TAKKOYSA-sınıflandırıcısı		4,8	**99,33**

Tablo 7.64.'de yer alan ortalama doğruluk etkinlik ölçütü değerleri incelendiğinde en iyi sonucu geliştirilen TAKKOYSA-sınıflandırıcısının verdiği görülmektedir. Çıkarılan kural sayıları dikkate alındığında ise geliştirilen algoritma 3,77 ile en az sayıda kural çıkaran CORE algoritması ile rekabet edebilecek düzeydedir. Çıkardığı kural sayısı 4,8 ile CORE algoritmasınınkine çok yakındır.

Algoritma, diğer yapay sinir ağlarından kural çıkaran algoritmalar ile karşılaştırıldığında ise tahminleyici doğruluklar bakımından daha iyi sonuç üretmiştir. Kural sayıları dikkate alındığında ise yapay sinir ağlarından kural çıkaran tüm algoritmaların Iris veri kümesi üzerinde birbirine çok yakın sayıda kural çıkardığı söylenebilir.

7.6.4. LBC Veri Kümesi İçin Karşılaştırma Sonuçları

Geliştirilen kural çıkarım algoritmasına 10-katlı çapraz geçerlilik testi uygulanması sonucu elde edilen sonuçların klasik makine öğrenme algoritmalarından NBTree, PART, DecisionTable ve C4.5 algoritmaları ile karşılaştırılması Tablo 7.65.'de verilmektedir. Her bir algoritmanın minimum, ortalama ve maksimum test doğruluğu, test doğruluğu standart sapması ve ortalama kural sayıları tabloda gösterilmektedir.

Tablo 7.65. LBC veri kümesi klasik sınıflandırıcılar ile karşılaştırma sonuçları.

Veri kümesi		NBTree	DT	PART	C4.5	Geliştirilen Algoritma
	Minimum	58,62	67,86	65,52	67,86	**89,29**
	Ortalama	72,75	75,53	70,99	75,16	**96,84**
LBC	Maksimum	86,21	82,76	72,41	86,21	**100**
	Standart sapma	6,96	4,64	**2,18**	5,10	4,23
	Ort. kural sayısı	**2,1**	14,6	17,7	9,1	29,3

Tablodan da görüldüğü gibi tahminleyici doğruluklar dikkate alındığında geliştirilen algoritma büyük bir farkla minimum, ortalama ve maksimum değerlerin üçünde de karşılaştırmaya tabi tutulan dört algoritmadan daha iyi sonuçlar bulmuştur. Standart sapma ise 4,23 ile diğer algoritmalara göre ortalama bir değerdir.

Çıkarılan ortalama kural sayısı dikkate alındığında ise, geliştirilen algoritma diğer algoritmalardan daha fazla kural içermektedir. Ancak karşılaştırmaya tabi tutulan

algoritmaların tahminleyici doğrulukları ile geliştirilen algoritmanın tahminleyici doğruluğu arasındaki farka dikkat edilecek olursa, kural sayısının artması da normaldir.

Mann Whitney U testi, elde edilen sonuçların daha iyi değerlendirilebilmesi ve algoritmaların elde ettikleri sonuçların arasında anlamlı bir farklılık olup olmadığının belirlenmesi için gerçekleştirilmiştir. Bu test, geliştirilen algoritma ile diğer her bir klasik makine öğrenme algoritması medyan değerleri arasındaki farkın anlamlı olup olmadığını test etmektedir. Tablo 7.66.'da algoritmaların medyan değerleri ve p değerleri gösterilmektedir.

Tablo 7.66. LBC veri kümesi için Mann Whitney U testi sonuçları.

	Etkinlik Ölçütü			
	Test Doğruluğu		Kural Sayısı	
Algoritma	Medyan	P değeri	Medyan	P değeri
TAKKOYSA-sınıflandırıcısı NBTree	100 73,71	0,0002	29 1	(-)0,0002
TAKKOYSA-sınıflandırıcısı Decision Table	100 75,43	0,0002	29 11	(-)0,0025
TAKKOYSA-sınıflandırıcısı PART	100 71,43	0,0002	29 17,5	(-)0,0005
TAKKOYSA-sınıflandırıcısı C4.5	100 73,71	0,0002	29 9,5	(-)0,0002

Tablo 7.66.'daki test doğruluğu-medyan değerleri incelendiğinde TAKKOYSA-sınıflandırıcısının medyan değerinin diğer dört algoritmanın medyan değerinden çok daha büyük olduğu görülür. Algoritma, alabileceği maksimum medyan değeri olan 100 değerini almıştır. Ancak medyanlar arasındaki farkın istatistiksel olarak anlamlı olup olmadığını söylemek için p değerlerine bakmak gerekir. Test doğruluğu p değerleri incelendiğinde geliştirilen algoritmanın elde ettiği tahminleyici doğruluklar arasında istatistiksel olarak anlamlı bir farklılık görülmektedir, bütün p değerleri 0,05'den küçük bir değer olan 0,0002 çıkmıştır. Geliştirilen algoritmanın medyan değerinin de daha büyük olması TAKKOYSA-sınıflandırıcısının diğer algoritmalardan istatistiksel olarak daha iyi sonuçlar üretebildiğini göstermektedir.

Tablo 7.66.'da kural sayısı sütununda yer alan (-) işaretli değerler, karşılaştırılan algoritmanın geliştirilen kural çıkarım algoritmasından daha iyi sonuç verdiğini ifade etmektedir. Çıkarılan kural sayıları dikkate alındığında, karşılaştırmada kullanılan dört algoritmanın da medyan değerlerinin geliştirilen algoritmadan küçük olduğu görülür. p değerleri karşılaştırmaya alınan dört algoritmanın da geliştirilen algoritmadan daha az sayıda kural ürettiğini ifade etmektedir ($p<0,05$).

Tablo 7.67. geliştirilen algoritma ile günümüz literatüründe yer alan kural tabanlı sınıflandırıcıların elde ettikleri tahminleyici doğruluklar ve ortalama kural sayılarının karşılaştırmasını vermektedir. Bazı algoritmaların ismi olmadığından bu algoritmalarda sadece referans verilmiştir.

Aynı şekilde, kural sayılarında bazı algoritmalara ait bilgi yer almamaktadır. Bunun nedeni ilgili algoritmanın ya her sınıf için sadece bir kural çıkarıyor olması veya kural sayısı bilgisinin bulunmamasıdır. *** ile gösterilen sınıflandırıcılar ise yapay sinir ağlarından kural çıkarım algoritmalarını ifade etmektedir.
Tablo 7.67. incelendiğinde ortalama doğruluk etkinlik ölçütünde en iyi sonucu TAKKOYSA-sınıflandırıcısının verdiği görülmektedir. Ancak çıkarılan kural sayısı diğer algoritmalara göre daha fazladır. Bunun nedeni daha önce de değinildiği gibi geliştirilen algoritmanın ikili diziler üzerinde çalışmasıdır.

Geliştirilen algoritma, Kahramanlı ve Allahverdi'nin (2009) yapay sinir ağlarından kural çıkarım algoritması ile karşılaştırıldığında, LBC veri kümesi üzerinde daha iyi tahminleyici doğruluklar elde ettiği görülür. Kural sayısı etkinlik ölçütü dikkate alındığında ise, geliştirilen TAKKOYSA-sınıflandırıcısı diğer algoritmanın yaklaşık yarısı kadar kural içeren bir kural kümesi ile daha yüksek doğruluk değeri elde etmiştir.

Tablo 7.67. LBC veri kümesinde TAKKOYSA-sınıflandırıcısının kural-tabanlı sınıflandırıcılar ile karşılaştırma sonuçları.

Veri Kümesi	**Algoritma**	**Referans**	**Kural Sayısı**	**Ortalama Doğruluk**
LBC	DOEA	Tan et al. (2006a)	5	78,91
	CORE	Tan et al. (2006b)	4,32	75,41
	MEPAR-miner	Baykasoğlu ve Özbakır (2007)	-	90,63
	-	Pitangui ve Zaverucha (2006)	**1**	76,14
	DCC	Tan et al. (2005)	5	76,16
	-	Wang ve Zhang (2005)	-	76,36
	Ant-Miner	Parpinelli et al. (2002b)	7,1	78,28
	ACO-Miner	Wang ve Feng (2004)	6,25	77,14
	Unordered Rule set Ant-Miner	Smaldon ve Freitas (2006)	6	60,31
	E-Ant-Miner	Ji et al. (2006)	5,9	75,37
	AntMiner+	Martens et al. (2007)	3,4	77,47
	-	Nalini ve Balasubramanie (2008)	7,5	79,9
	CPSO	Sousa et al. (2004)	5	74,2
	PSO/ACO2	Holden ve Freitas (2007)	-	74,16
	GP	Bojarczuk et al. (2004)	-	71.8
	TACO-miner	Thangavel ve Jaganathan, 2007	5,73	77,76
	***Rule extraction from adaptive ANN using AIS	Kahramanlı ve Allahverdi (2009)	59	92,31
	TAKKOYSA-sınıflandırıcısı		29,3	**96,84**

7.6.5. Monks-2 Veri Kümesi İçin Karşılaştırma Sonuçları

Tablo 7.68. Monks-2 veri kümesinde tek eğitim ve test kümeleri olmasından dolayı, aynı eğitim ve test kümelerine algoritmanın 10 kez uygulanması ile elde edilen sonuçların, ele alınan klasik makine öğrenme algoritmaları ile karşılaştırılmasını

vermektedir. Karşılaştırmalar minimum, ortalama ve maksimum test doğruluğu, test doğruluğu standart sapması ve ortalama kural sayıları dikkate alınarak gerçekleştirilmiştir.

Tablo 7.68. Monks-2 veri kümesi klasik sınıflandırıcılar ile karşılaştırma sonuçları.

Veri kümesi		NBTree	DT	PART	C4.5	Geliştirilen Algoritma
	Minimum	67,176	67,176	67,557	62,977	**97,45**
	Ortalama	67,176	67,176	75,992	71,832	**99,07**
Monks-2	Maksimum	67,176	67,176	83,588	85,496	**100**
	Standart sapma	**0**	**0**	5,889	5,835	0,78
	Ort. kural sayısı	**1**	**1**	15	19,4	15,2

Tablodan da görüldüğü gibi tahminleyici doğruluklar dikkate alındığında geliştirilen algoritma büyük bir farkla minimum, ortalama ve maksimum değerlerin üçünde de karşılaştırmaya tabi tutulan dört algoritmadan daha iyi sonuçlar bulmuştur. Diğer algoritmalar ile geliştirilen kural çıkarım algoritması arasında % 30 gibi çok büyük bir doğruluk farkı vardır.

Çıkarılan ortalama kural sayısı dikkate alındığında ise geliştirilen algoritma, 15,2 kural ile diğer algoritmalara göre ortalama sayıda kural içeren bir kural kümesi üretmiştir. % 30'luk bir doğruluk farkıyla bu sayıda kural içeren bir kural kümesi üretmek, algoritmanın başarısını göstermektedir.

Elde edilen sonuçların daha iyi değerlendirilebilmesi ve aralarında anlamlı bir farklılık olup olmadığının belirlenmesi için gerçekleştirilen Mann Whitney U testi sonuçları Tablo 7.69.'da gösterilmektedir. Tabloda algoritmaların medyan ve p değerleri yer almaktadır.

Tablo 7.69. Monks-2 veri kümesi için Mann Whitney U testi sonuçları.

	Etkinlik Ölçütü			
	Test Doğruluğu		Kural Sayısı	
Algoritma	Medyan	P değeri	Medyan	P değeri
TAKKOYSA-sınıflandırıcısı	99,074	-	16	-
NBTree	-		-	
TAKKOYSA-sınıflandırıcısı	99,074	-	16	-
Decision Table	-		-	
TAKKOYSA-sınıflandırıcısı	99,074	0,0002	16	**1,0000**
PART	75,955		15	
TAKKOYSA-sınıflandırıcısı	99,074	0,0002	16	**0,1306**
C4.5	71,756		19,5	

Tablo 7.69.'daki test doğruluğu-medyan değerleri incelendiğinde geliştirilen algoritmanın medyan değerinin PART ve C4.5 medyan değerlerinden büyük olduğu görülür. Ancak medyanlar arasındaki farkın istatistiksel olarak anlamlı olup olmadığını söylemek için p değerlerine bakmak gerekir. Test doğruluğu p değerleri incelendiğinde TAKKOYSA-sınıflandırıcısının elde ettiği tahminleyici doğruluklar arasında istatistiksel olarak anlamlı bir farklılık görülmektedir ($p<0,05$). Geliştirilen algoritmanın medyan değerinin PART ve C4.5 algoritmalarından daha büyük olması geliştirilen algoritmanın bu iki algoritmadan istatistiksel olarak daha iyi sonuçlar üretebildiğini göstermektedir. NBTree ve DecisionTable algoritmaları her 10 çalıştırmada da aynı sonuçları üretmiştir. Sonuçlar arasında standart sapma 0 olduğu için Mann Whitney U testi gerçekleştirilememiştir. Ancak ortalama ve standart sapma istatistiksel ölçütlere göre algoritmaların elde ettikleri sonuçları karşılaştırmak mümkündür. Geliştirilen algoritma % 99,07 gibi yüksek bir ortalama test doğruluğu ile ortalama % 67,176 test doğruluğuna sahip olan diğer algoritmalardan çok daha iyi sonuçlar üretmiştir.

NBTree ve DecisionTable algoritmaları tek bir kural ile % 67,176 test doğruluğu elde etmişlerdir. Geliştirilen algoritma ise yaklaşık % 100 doğruluğu 15,2 kural ile elde etmiştir. Kural sayıları arasındaki matematiksel fark istatistiksel farka da işaret etmektedir. Fakat test doğruluğunun aksine burada diğer algoritmalar daha iyi sonuç vermiştir.

Tablodaki koyu işaretlenmiş değerler, ilgili veri kümesinde, karşılaştırılan algoritmaların medyan sonuçlarının birbirinden farklı olduğunu ispatlamanın istatistiksel olarak mümkün olmadığını göstermektedir. Kural sayısı etkinlik ölçütünde PART ve C4.5 algoritmaları ile geliştirilen algoritmanın elde ettiği kural sayıları arasında istatistiksel olarak bir anlamlılık yoktur, çünkü p değeri önemlilik düzeyinden büyüktür (p>0,005). Geliştirilen algoritma, bu iki algoritmayla yakın sayıda kuralla çok daha yüksek doğruluklar elde edebilmektedir.

Tablo 7.70. TAKKOYSA-sınıflandırıcısı ile günümüz literatüründe yer alan kural tabanlı sınıflandırıcıların elde ettikleri tahminleyici doğruluklar ve ortalama kural sayılarının karşılaştırmasını vermektedir. Tabloda kural sayılarında bazı algoritmalara ait bilgi yer almamaktadır. Bunun nedeni ilgili algoritmanın ya her sınıf için sadece bir kural çıkarıyor olması veya kural sayısı bilgisinin bulunmamasıdır. *** ile gösterilen sınıflandırıcılar ise yapay sinir ağlarından kural çıkarım algoritmalarını ifade etmektedir.

Tablo 7.70. Monks-2 veri kümesinde TAKKOYSA-sınıflandırıcısının kural-tabanlı sınıflandırıcılar ile karşılaştırma sonuçları.

Veri Kümesi	**Algoritma**	**Referans**	**Kural Sayısı**	**Ortalama Doğruluk**
Monks-2	INPGA	Dehuri ve Mall (2006)	**7**	76,65
	MEPAR-miner	Baykasoğlu ve Özbakır (2007)	-	95,83
	C4.5/GA-small	Carvalho ve Freitas (2004)	-	96,93
	C4.5/GA-large-SN	Carvalho ve Freitas (2004)	-	96,77
	TFPC	Coenen ve Leng (2007)	39,7	77,8
	CMAR	Coenen ve Leng (2007)	298,6	88,3
	CBA	Coenen ve Leng (2007)	78,4	90,1
	*** -	Arbatlı ve Akın (1996)	21	60
	TAKKOYSA-sınıflandırıcısı		39,7	**99,98**

Tablo 7.70. incelendiğinde ortalama doğruluk bakımından en iyi sonucu geliştirilen algoritmanın verdiği görülmektedir. Çıkarılan kural sayıları dikkate alındığında ise ikinci en iyi algoritmadır.

Algoritma, Arbatlı ve Akın'ın (1996) yapay sinir ağlarından kural çıkarım algoritması ile karşılaştırıldığında ise geliştirilen algoritma %40'lık bir farkla daha iyi sonuçlar elde etmiştir. Çıkarılan kural sayısı dikkate alındığında, daha fazla kural içermektedir ancak doğruluklar arasındaki fark dikkate alınırsa kural sayısının artması normaldir.

7.6.6. Nursery Veri Kümesi İçin Karşılaştırma Sonuçları

Nursery veri kümesi 12.960 örnek içeren büyük bir veri kümesidir. Bu veri kümesinde, geliştirilen algoritmaya 10-katlı çapraz geçerlilik testi uygulanması ile elde edilen sonuçların değerlendirmeye alınan klasik makine algoritmaları ile karşılaştırması Tablo 7.71.'de verilmektedir. Tablo, minimum, ortalama ve maksimum test doğruluğu, test doğruluğu standart sapması ve ortalama kural sayılarını içermektedir.

Tablo 7.71. Nursery veri kümesi klasik sınıflandırıcılar ile karşılaştırma sonuçları.

Veri kümesi		NBTree	DT	PART	C4.5	Geliştirilen Algoritma
Nursery	Minimum	96,53	94,06	98,69	96,14	**99,92**
	Ortalama	97,49	94,78	99,21	97,05	**99,98**
	Maksimum	98,15	95,60	99,61	97,69	**100**
	Standart sapma	0,5	0,6	0,28	0,47	**0,03**
	Ort. kural sayısı	149,1	1053	191,9	352,9	**39,7**

Tabloda verilen minimum, ortalama ve maksimum doğruluklar incelendiğinde, geliştirilen algoritmanın en yüksek doğrulukları bulduğu görülmektedir. Ancak diğer algoritmalar ile aradaki doğruluk farkı çok değildir. Doğruluktaki standart sapma değeri en düşük olan algoritma da yine geliştirilen algoritmadır. Bu da ele alınan eğitim ve test kümesi değişse de kural çıkarım algoritmasının kaliteli çözümler üretebildiğini göstermektedir.

Çıkarılan ortalama kural sayısı dikkate alındığında ise geliştirilen algoritma diğer algoritmalardan çok daha iyi kural kümeleri bulmuştur. Geliştirilen TAKKOYSA-sınıflandırıcısının kural kümesindeki kural sayısı ortalama 39,7 iken, ikinci en düşük kural sayısına sahip algoritma 149, 1 gibi büyük bir değerle NBTree algoritmasıdır. En yüksek sayıda kuralı ise 352,9 gibi bir kural ile C4.5 algoritması bulmuştur.

Elde edilen sonuçların daha iyi değerlendirilebilmesi ve aralarında anlamlı bir farklılık olup olmadığının belirlenmesi için gerçekleştirilen Mann Whitney U testi sonuçları Tablo 7.72.'de gösterilmektedir. Tabloda algoritmaların medyan değerleri ve p değerleri gösterilmektedir.

Tablo 7.72. Nursery veri kümesi için Mann Whitney U testi sonuçları.

	Etkinlik Ölçütü			
	Test Doğruluğu		Kural Sayısı	
Algoritma	Medyan	P değeri	Medyan	P değeri
TAKKOYSA-sınıflandırıcısı NBTree	100 97,49	0,0002	39 147,5	0,0002
TAKKOYSA-sınıflandırıcısı Decision Table	100 94,71	0,0002	39 810	0,0002
TAKKOYSA-sınıflandırıcısı PART	100 99,23	0,0002	39 189,5	0,0002
TAKKOYSA-sınıflandırıcısı C4.5	100 96,99	0,0002	39 354,5	0,0002

Tablo 7.72.'deki test doğruluğu-medyan değerleri incelendiğinde geliştirilen algoritmanın medyan değerinin diğer dört algoritmanın medyan değerinden büyük olduğu görülür. Ancak medyanlar arasındaki farkın istatistiksel olarak anlamlı olup olmadığını söylemek için p değerlerine bakmak gerekir. Eğer p-değeri α=0,05 önemlilik değerinden küçükse, karşılaştırılan algoritmaların medyan değerleri arasında istatistiksel olarak anlamlı bir farklılık var demektir. Test doğruluğu p değerleri incelendiğinde geliştirilen algoritmanın elde ettiği tahminleyici doğruluklar arasında istatistiksel olarak anlamlı bir farklılık görülmektedir ($p<0,05$). Geliştirilen algoritmanın medyan değerinin

daha yüksek olması, TAKKOYSA-sınıflandırıcısının diğer algoritmalardan istatistiksel olarak daha iyi sonuçlar üretebildiğini göstermektedir.

Çıkarılan kural sayıları dikkate alındığında, geliştirilen kural çıkarım algoritmasının medyan değerinin karşılaştırmada kullanılan dört algoritmanın da medyan değerinden küçük olduğu görülür. p değerleri incelendiğinde geliştirilen algoritmanın istatistiksel olarak diğer algoritmalardan daha az sayıda kural ürettiği söylenebilir ($p<0,05$).

Tablo 7.73. geliştirilen algoritma ile günümüz literatüründe yer alan kural tabanlı sınıflandırıcıların elde ettikleri tahminleyici doğruluklar ve ortalama kural sayılarının karşılaştırmasını vermektedir. Tabloda, kural sayısı sütununda bazı algoritmalara ait bilgi yer almamaktadır. Bunun nedeni ilgili algoritmanın ya her sınıf için sadece bir kural çıkarıyor olması veya kural sayısı bilgisinin bulunmamasıdır.

Tablo 7.73. Nursery veri kümesinde TAKKOYSA-sınıflandırıcısının kural-tabanlı sınıflandırıcılar ile karşılaştırma sonuçları.

Veri Kümesi	**Algoritma**	**Referans**	**Kural Sayısı**	**Ortalama Doğruluk**
Nursery	INPGA	Dehuri ve Mall (2006)	**7**	76,65
	MEPAR-miner	Baykasoğlu ve Özbakır (2007)	-	95,83
	C4.5/GA-small	Carvalho ve Freitas (2004)	-	96,93
	C4.5/GA-large-SN	Carvalho ve Freitas (2004)	-	96,77
	TFPC	Coenen ve Leng (2007)	39,7	77,8
	CMAR	Coenen ve Leng (2007)	298,6	88,3
	CBA	Coenen ve Leng (2007)	78,4	90,1
	TAKKOYSA-sınıflandırıcısı		39,7	**99,98**

Tablo 7.73. incelendiğinde ortalama doğruluk etkinlik ölçütünde en iyi sonucu geliştirilen algoritmanın verdiği görülmektedir. Çıkarılan kural sayıları dikkate alındığında ise en az sayıda kuralı vermese de ikinci sırada yer almaktadır. Bu da algoritmanın elde ettiği test doğruluğu ile kural sayısının diğer algoritmalara göre daha iyi olduğunu göstermektedir.

Algoritma, Nursery veri kümesinde diğer yapay sinir ağlarından kural çıkaran algoritmalar ile karşılaştırılamamıştır. Bunun nedeni literatürde bu veri kümesini kullanan bir kural çıkarım algoritmasının bulunamamasıdır.

7.6.7. Pima Veri Kümesi İçin Karşılaştırma Sonuçları

Tablo 7.74. geliştirilen algoritmaya 10-katlı çapraz geçerlilik testi uygulanması ile Pima veri kümesinde elde edilen sonuçların NBTree, PART, DecisionTable ve C4.5 algoritmaları ile karşılaştırılmasını vermektedir. Karşılaştırmalar minimum, ortalama ve maksimum test doğruluğu, test doğruluğu standart sapması ve ortalama kural sayıları dikkate alınarak gerçekleştirilmiştir.

Tablo 7.74. Pima veri kümesi klasik sınıflandırıcılar ile karşılaştırma sonuçları.

Veri kümesi		NBTree	DT	PART	C4.5	Geliştirilen Algoritma
	Minimum	66,23	67,53	64,94	64,94	**97,4**
	Ortalama	74,36	72,27	74,49	73,84	**99,61**
Pima	Maksimum	85,53	77,92	79,22	81,82	**100**
	Standart sapma	6,68	3,56	5,09	5,66	**0,88**
	Ort. kural sayısı	**3,2**	51,5	7,5	19,2	31,5

Tablo 7.74.'den de görüldüğü gibi geliştirilen TAKKOYSA-sınıflandırıcısı ortalama % 99,61 doğrulukta kurallar üretebilmektedir. Klasik makine öğrenme algoritmaları ile karşılaştırıldığında ise aralarında % 25'lik bir doğruluk farkı bulunmaktadır. On katlı çapraz doğrulamada sonuçlar arasındaki en düşük standart sapma da geliştirilen algoritmaya aittir.

Çıkarılan ortalama kural sayısı dikkate alındığında ise geliştirilen algoritma NBTree, PART ve C4.5 algoritmalarından daha fazla kural içermektedir. DecisionTable ise en fazla kural içeren algoritmadır. Geliştirilen algoritmanın kural sayısı, % 25'lik bir doğruluk artışıyla beş algoritma arasında ortalarda yer almaktadır. Bu da geliştirilen algoritmanın diğer algoritmalardan daha iyi sonuçlar elde edebildiğini göstermektedir.

Elde edilen sonuçların daha iyi değerlendirilebilmesi ve aralarında anlamlı bir farklılık olup olmadığının test edilmesi için gerçekleştirilen Mann Whitney U testi sonuçları Tablo 7.75.'de yer almaktadır. Bu test, geliştirilen algoritma ve diğer her bir klasik makine öğrenme algoritması medyan değerleri arasındaki farkın anlamlı olup olmadığını ölçmekte kullanılmıştır. Tabloda algoritmaların medyan ve p değerleri gösterilmektedir.

Tablo 7.75. Pima veri kümesi için Mann Whitney U testi sonuçları.

	Etkinlik Ölçütü			
	Test Doğruluğu		Kural Sayısı	
Algoritma	Medyan	P değeri	Medyan	P değeri
TAKKOYSA-sınıflandırıcısı NBTree	100 73,38	0,0002	31 3	(-)0,0002
Geliştirilen Algoritma Decision Table	100 72,08	0,0002	31 52,5	0,0376
TAKKOYSA-sınıflandırıcısı PART	100 75,17	0,0002	31 7	(-)0,0002
TAKKOYSA-sınıflandırıcısı C4.5	100 75,32	0,0002	31 18	(-)0,0004

Tablo 7.75.'de yer alan test doğruluğu-medyan değerleri incelendiğinde geliştirilen algoritmanın medyan değerinin diğer dört makine öğrenme algoritmasının medyan değerinden maksimum alabileceği 100 değeri ile daha yüksek olduğu görülür. Ancak medyanlar arasındaki farkın istatistiksel olarak anlamlı olup olmadığını söylemek için p değerlerine bakmak gerekir. Test doğruluğu p değerleri incelendiğinde geliştirilen algoritmanın elde ettiği test doğrulukları arasında istatistiksel olarak anlamlı bir farklılık görülmektedir ($p<0,05$). Bu farklılık, medyan değerinin diğer algoritmalardan yüksek olması nedeniyle geliştirilen algoritma yönündedir. Algoritma, istatistiksel olarak diğer algoritmalardan daha yüksek doğrulukta çözümler üretebilmektedir.

Tablo 7.57.'de (-) işaretli değerler, karşılaştırılan algoritmanın geliştirilen kural çıkarım algoritmasından daha iyi sonuç verdiğini ifade etmektedir. Çıkarılan kural sayıları dikkate alındığında, NBTree, PART ve C4.5 algoritmalarının medyan değerlerinin

geliştirilen algoritmadan küçük olduğu görülür. DecisionTable algoritmasının medyan değeri ise daha yüksektir. p değerleri incelendiğinde DecisionTable algoritması dışında kalan üç algoritmanın istatistiksel olarak geliştirilen algoritmadan daha az sayıda kural ürettiği söylenebilir ($p<0,05$). Geliştirilen algoritma $p<0,005$ olduğu için DecisionTable algoritmasından daha az sayıda kural içeren kural kümeleri üretebilmektedir.

Tablo 7.76. geliştirilen algoritma ile günümüz literatüründe yer alan kural tabanlı sınıflandırıcıların elde ettikleri tahminleyici doğruluklar ve ortalama kural sayılarının karşılaştırmasını vermektedir. Tabloda, kural sayılarında bazı algoritmalara ait bilgi yer almamaktadır. Bunun nedeni ilgili algoritmanın ya her sınıf için sadece bir kural çıkarıyor olması veya kural sayısı bilgisinin bulunmamasıdır. *** ile gösterilen sınıflandırıcılar ise yapay sinir ağlarından kural çıkarım algoritmalarını ifade etmektedir.

Tablo 7.76. Pima veri kümesinde TAKKOYSA-sınıflandırıcısının kural-tabanlı sınıflandırıcılar ile karşılaştırma sonuçları.

Veri Kümesi	**Algoritma**	**Referans**	**Kural Sayısı**	**Ortalama Doğruluk**
Pima	Ant-Miner with Imp. Quic k Red.	Jaganathan et al. (2007)	11,3	76,58
	GARC	Chen et al. (2006)	**6**	73,83
	TFPC	Coenen ve Leng (2007)	25	74,4
	CMAR	Coenen ve Leng (2007)	87,9	74,4
	CBA	Coenen ve Leng (2007)	26,6	75
	RMR	Thabtah ve Cowling (2007)	65	78,05
	-	Terson ve Zhang (2007)	-	68,6
	***CGA	Hruschka ve Ebecken (2006)	5	66,23
	***G-REX oracle	Johansson vd., 2006	-	75,9
	TAKKOYSA-sınıflandırıcısı		31,5	**99,61**

Ortalama doğruluk etkinlik ölçütünde en iyi sonucu % 20'lik bir doğruluk farkıyla geliştirilen TAKKOYSA-sınıflandırıcısının verdiği, Tablo 7.76.'dan görülmektedir.

Çıkarılan kural sayıları dikkate alındığında ise geliştirilen algoritma diğer kural tabanlı sınıflandırıcılar ile rekabet edecek sayıda kural bulmuştur. Bulunan kural sayısı en az olmasa da diğer sınıflandıcıların kural sayılarına göre orta değerlerde yer almaktadır. Pima veri kümesinde geliştirilen algoritma kaliteli çözümler üretmiştir.

Algoritma, diğer yapay sinir ağlarından kural çıkaran algoritmalar ile karşılaştırıldığında ise tahminleyici doğruluklar arasında önemli bir fark görülmektedir. Diğer algoritmalar % 60-70'lerde doğruluk elde ederken, geliştirilen algoritma % 99,61 doğruluk elde etmiştir. Geliştirilen algoritmanın kural sayısı ise daha yüksektir. Ancak kural indirgemede uygunluk fonksiyonu üzerinde değişiklik yapılarak bu kural sayısı düşürülebilir.

7.6.8. Spect-heart Veri Kümesi İçin Karşılaştırma Sonuçları

Geliştirilen kural çıkarım algoritmasının Spect-heart veri kümesinde aynı eğitim ve test kümesiyle 10 kez çalıştırılması ile elde edilen sonuçların NBTree, PART, DecisionTable ve C4.5 algoritmaları ile karşılaştırılması Tablo 7.77.'de gösterilmektedir.. Minimum, ortalama ve maksimum test doğruluğu, test doğruluğu standart sapması ve ortalama kural sayıları tabloda yer almaktadır.

Tablo 7.77. Spect-heart veri kümesi klasik sınıflandırıcılar ile karşılaştırma sonuçları.

Veri kümesi		NBTree	DT	PART	C4.5	Geliştirilen Algoritma
	Minimum	59,893	59,358	58,065	59,14	**90,37**
	Ortalama	66,864	67,663	64,719	65,735	**94,76**
Spect-heart	Maksimum	70,43	73,262	70,053	74,866	**98,4**
	Standart sapma	3,477	4,291	3,853	5,22	**2,52**
	Ort. kural sayısı	**2,1**	14,6	17,7	9,1	29,3

Tahminleyici doğruluklar dikkate alındığında, geliştirilen algoritma karşılaştırmaya tabi tutulan diğer dört algoritmadan da minimum, ortalama ve maksimum test doğruluğunda daha iyi sonuçlar elde etmiştir. Programın aynı eğitim ve test kümesinde on kez çalıştırılması sonucu elde edilen doğruluk standart sapması arasındaki en düşük değer yine geliştirilen algoritmaya aittir.

Çıkarılan ortalama kural sayısı dikkate alındığında ise geliştirilen algoritma diğer algoritmalara göre daha fazla sayıda kural içeren kural kümeleri üretmiştir. En az kural, 2,1 ile NBTRee algoritmasına aittir, ancak bu algoritmanın tahminleyici doğruluğu ile geliştirilen algoritmanın elde ettiği doğruluk arasında büyük bir fark vardır.

Elde edilen sonuçların aralarında istatistiksel olarak anlamlı bir farklılık olup olmadığının değerlendirilmesi için gerçekleştirilen Mann Whitney U testi sonuçları Tablo 7.78.'de yer almaktadır. Tabloda algoritmaların medyan ve p değerleri gösterilmektedir.

Tablo 7.78. Spect-heart veri kümesi için Mann Whitney U testi sonuçları.

	Etkinlik Ölçütü			
	Test Doğruluğu		Kural Sayısı	
Algoritma	Medyan	P değeri	Medyan	P değeri
TAKKOYSA-sınıflandırıcısı NBTree	94,92 67,915	0,0002	18 4	(-)0,0002
TAKKOYSA-sınıflandırıcısı Decision Table	94,92 68,095	0,0002	18 18,5	**0,8501**
TAKKOYSA-sınıflandırıcısı PART	94,92 64,885	0,0002	18 8,5	(-)0,0003
TAKKOYSA-sınıflandırıcısı C4.5	94,92 65,245	0,0002	18 8,5	(-)0,0002

Tabloda verilen test doğruluğu-medyan değerleri incelendiğinde geliştirilen algoritmanın medyan değerinin karşılaştırmada kullanılan dört algoritmanın medyan değerinden daha büyük olduğu görülür. Ancak medyanlar arasındaki farkın istatistiksel olarak anlamlı olup olmadığını söylemek için medyan değerleri yeterli değildir, p değerlerini de incelemek gerekir. Test doğruluğu p değerleri incelendiğinde geliştirilen algoritmanın elde ettiği tahminleyici doğruluklar arasında istatistiksel olarak anlamlı bir farklılık görülmektedir ($p<0,05$). Bu farklılık geliştirilen algoritmanın diğer algoritmalardan daha yüksek doğrulukta kurallar üretebildiğini ifade etmektedir.

Tablo 7.78.'de (-) işaretli değerler, karşılaştırılan algoritmanın geliştirilen kural çıkarım algoritmasından daha iyi sonuç verdiğini ifade etmektedir. Çıkarılan kural sayıları

dikkate alındığında, DecisionTable algoritması dışında kalan üç algoritmanın medyan değerlerinin geliştirilen algoritmadan küçük olduğu görülür. p değerleri incelendiğinde bu üç algoritmanın istatistiksel olarak geliştirilen algoritmadan daha az sayıda kural ürettiği söylenebilir ($p<0,05$).

Tablodaki koyu işaretlenmiş değerler, ilgili veri kümesinde, karşılaştırılan algoritmaların medyan sonuçlarının birbirinden farklı olduğunu ispatlamanın istatistiksel olarak mümkün olmadığını ifade etmektedir. Çünkü bu değerler 0,05 önemlilik düzeyinden büyüktür. Kural sayısı etkinlik ölçütünde DecisionTable algoritması ile geliştirilen algoritmanın elde ettiği kural sayıları arasında istatistiksel olarak bir anlamlılık yoktur, çünkü p değeri önemlilik düzeyinden büyüktür ($p>0,005$).

Tablo 7.79. geliştirilen algoritma ile günümüz literatüründe yer alan kural tabanlı sınıflandırıcıların elde ettikleri tahminleyici doğruluklar ve ortalama kural sayılarının karşılaştırılmasını vermektedir. Tabloda, kural sayılarında bazı algoritmalara ait bilgi yer almamaktadır. Bunun nedeni ilgili algoritmanın ya her sınıf için sadece bir kural çıkarıyor olması veya kural sayısı bilgisinin bulunmamasıdır.

Tablo 7.79. Spect-heart veri kümesinde TAKKOYSA-sınıflandırıcısının kural tabanlı sınıflandırıcılar ile karşılaştırma sonuçları.

Veri Kümesi	**Algoritma**	**Referans**	**Kural Sayısı**	**Ortalama Doğruluk**
Spect-heart	CLIP3	Kurgan vd., 2001	2	79,68
	-	Terson ve Zhang (2007)	-	82,2
	TAKKOYSA-sınıflandırıcısı		17,2	94,76

Tablo 7.79. incelendiğinde ortalama doğruluk etkinlik ölçütünde en iyi sonucu diğer algoritmalardan büyük bir farkla geliştirilen algoritmanın verdiği görülmektedir. Kural sayısı olarak sadece veri kümesinin sahibi olan Kurgan vd. (2001)'in sonuçları yer almaktadır. Kurgan vd.'nin kural sayısı geliştirilen algoritmanınkine göre oldukça düşüktür, ancak test doğrulukları da düşüktür.

Literatürde Spect-heart veri kümesi üzerinde analiz yapan herhangi bir YSA'dan kural çıkarım algoritmasına rastlanmadığından bu veri kümesi için kural çıkarım algoritmaları arası değerlendirme yapılamamıştır.

7.6.9. Tic-Tac-Toe Veri Kümesi İçin Karşılaştırma Sonuçları

Tablo 7.80. geliştirilen algoritmaya 10-katlı çapraz geçerlilik testi uygulanması sonucu elde edilen sonuçların dört farklı klasik makine öğrenme algoritması ile karşılaştırmasıını vermektedir. Karşılaştırmalar minimum, ortalama ve maksimum test doğruluğu, test doğruluğu standart sapması ve ortalama kural sayıları dikkate alınarak gerçekleştirilmiştir.

Tablo 7.80. Tic-Tac-Toe veri kümesi klasik sınıflandırıcılar ile karşılaştırma sonuçları.

Veri kümesi		NBTree	DT	PART	C4.5	Geliştirilen Algoritma
	Minimum	75	75	87,5	75,789	**100**
	Ortalama	84,863	78,913	94,887	85,069	**100**
Tic-Tac-Toe	Maksimum	90,625	82,292	97,917	90,526	**100**
	Standart sapma	4,539	2,139	3,159	4,492	**0**
	Ort. kural sayısı	45,4	357,4	39,8	90,6	**16,1**

Geliştirilen yapay sinir ağlarından kural çıkarım algoritması Tic-tac-toe veri kümesinde 10 katlı çapraz doğrulamadaki her bir katta % 100 doğruluk elde etmiştir. Dolayısıyla algoritmanın test doğruluğu standart sapması sıfırdır. Diğer algoritmaların hiç biri maksimum doğrulukta dahi bu değere ulaşamamıştır. Geliştirilen algoritma elde edilebilecek maksimum doğrulukla diğer algoritmalardan çok daha kaliteli çözümler üretebilmektedir.

Çıkarılan ortalama kural sayısı dikkate alındığında ise geliştirilen algoritma diğer algoritmalardan daha az sayıda kural içermektedir. Kural sayısı olarak kendisine en yakın algoritmanın kural sayısı bile geliştirilen algoritmanın elde ettiği ortalama kural sayısının iki katından fazladır. Sonuç olarak geliştirilen kural çıkarım algoritması,

maksimum doğruluk ile diğer algoritmalara göre çok daha az sayıda kural ile doğruluğu yüksek kümeler üretebilmektedir.

Elde edilen sonuçların daha iyi değerlendirilebilmesi ve aralarında anlamlı bir farklılık olup olmadığının belirlenmesi için gerçekleştirilen Mann Whitney U testi sonuçları Tablo 7.81.'de gösterilmektedir. Bu test ile α=0,05 önemlilik düzeyinde, geliştirilen algoritma ve diğer her bir klasik makine öğrenme algoritması medyan değerleri arasındaki farkın anlamlı olup olmadığı test edilmiştir. Tabloda algoritmaların medyan ve p değerleri gösterilmektedir.

Tablo 7.81. Tic-Tac-Toe veri kümesi için Mann Whitney U testi sonuçları.

	Etkinlik Ölçütü			
	Test Doğruluğu		Kural Sayısı	
Algoritma	Ortalama	Std.Sapma	Medyan	P değeri
TAKKOYSA-sınıflandırıcısı	100	0	16,5	0,0002
NBTree	84,86	4,54	46	
TAKKOYSA-sınıflandırıcısı	100	0	16,5	0,0002
Decision Table	78,91	2,14	175	
TAKKOYSA-sınıflandırıcısı	100	0	16,5	0,0002
PART	94,89	3,16	40	
TAKKOYSA-sınıflandırıcısı	100	0	16,5	0,0002
C4.5	85,07	4,49	90	

Geliştirilen algoritma 10-katlı çapraz doğrulamanın her katında % 100 doğruluk elde ettiği için doğruluk standart sapması 0'dır. Bu nedenle test doğruluğunda Mann Whitney U testi gerçekleştirilememiştir. Karşılaştırmada kullanılan algoritmalar ile geliştirilen algoritmaların elde ettikleri test doğrulukları arasında anlamlı bir farklılık olup olmadığı, algoritmaların ortalama ve standart sapma değerlerine göre yapılmıştır. Tablo 7.81.'in test doğruluğu sütunu ortalama ve standart sapma değerlerini içermektedir. Bu değerler incelendiğinde geliştirilen algoritma ile diğer algoritmalar arasında önemli bir doğruluk farkı olduğu görülmektedir. Algoritmanın 0 standart sapma ile % 100 doğruluk elde etmesi, algoritmanın performansını göstermektedir.

Çıkarılan kural sayıları dikkate alındığında, geliştirilen algoritmanın medyan değerinin karşılaştırmada kullanılan dört algoritmanın medyan değerlerinden küçük olduğu görülmektedir. p değerleri incelendiğinde, karşılaştırmada kullanılan diğer algoritmalara göre geliştirilen algoritmanın az sayıda kuralla yüksek doğrulukta kural kümeleri üretebildiği söylenebilir ($p<0,05$).

Tablo 7.82. geliştirilen algoritma ile günümüz literatüründe yer alan kural tabanlı sınıflandırıcıların elde ettikleri tahminleyici doğruluklar ve ortalama kural sayılarının karşılaştırmasını vermektedir.

Tablo 7.82. incelendiğinde ortalama doğruluk etkinlik ölçütünde en iyi sonucu GARC, RMR ve CBA algoritmalarının yanında geliştirilen TAKKOYSA-sınıflandırıcısının verdiği görülmektedir. Dört algoritma da maksimum doğruluk değerini elde etmiştir. Çıkarılan kural sayısını doğruluklar eşit olduğu için, bu üç algoritma ile karşılaştırmak daha mantıklıdır. Kural sayısı bakımından en az kuralı 9 kural ile CBA algritması elde etmiştir. İkinci sırada ise 16,1 kural ile geliştirilen algoritma yer almaktadır. Dolayısıyla algoritma bu veri kümesinde de iyi sonuçlar vermiştir.

Algoritma, diğer bir YSA üzerinde çalışan G-REX oracle algoritması ile karşılaştırıldığında ise tahminleyici doğruluklar arasında büyük bir fark görülmektedir. G-REX oracle algoritmasının kural sayısı verilmediği için kural sayıları karşılaştırılamamıştır.

Tablo 7.82. Tic-Tac-Toe veri kümesinde TAKKOYSA-sınıflandırıcısının kural-tabanlı sınıflandırıcılar ile karşılaştırma sonuçları.

Veri Kümesi	**Algoritma**	**Referans**	**Kural Sayısı**	**Ortalama Doğruluk**
Tic-Tac-Toe	MEPAR-miner	Baykasoğlu ve Özbakır (2007)	-	94,47
	-	Wang ve Zhang (2005)	-	97,84
	Ant-Miner	Parpinelli et al. (2002b)	8,5	73,04
	ACO-Miner	Wang ve Feng (2004)	6,47	98,43
	Ant-Miner3	Liu et al. (2003)	18,58	76,58
	Unordered Rule set Ant-Miner	Smaldon ve Freitas (2006)	**6,3**	64,39
	AntMiner+	Martens et al. (2007)	9	99,75
	-	Nalini ve Balasubramanie (2008)	7,16	75,01
	PSO/ACO2	Holden ve Freitas (2007)	-	98,64
	GARC	Chen et al. (2006)	26	**100**
	RMR	Thabtah ve Cowling (2007)	26	**100**
	TFPC	Coenen ve Leng (2007)	67,2	67,1
	CMAR	Coenen ve Leng (2007)	362	93,5
	CBA	Coenen ve Leng (2007)	9	**100**
	E-Ant-Miner	Ji et al. (2006)	7,6	74,33
	TACO-miner	Thangavel ve Jaganathan, 2007	5,57	71,87
	***G-REX oracle	Johansson vd., 2006	-	81,6
	TAKKOYSA-sınıflandırıcısı		16,1	**100**

7.6.10. Vote Veri Kümesi İçin Karşılaştırma Sonuçları

Tablo 7.83 geliştirilen algoritmaya 10-katlı çapraz geçerlilik testi uygulanması ile elde edilen sonuçların NBTree, PART, DecisionTable ve C4.5 algoritmaları ile karşılaştırmasını vermektedir. Minimum, ortalama ve maksimum test doğrulukları, standart sapmalar ve ortalama kural sayıları her algoritma için tabloda gösterilmektedir.

Tablo 7.83. Vote veri kümesi klasik sınıflandırıcılar ile karşılaştırma sonuçları.

Veri kümesi		NBTree	DT	PART	C4.5	Geliştirilen Algoritma
	Minimum	88,372	84,091	84,091	88,936	**95,35**
	Ortalama	94,255	95,412	95,063	96,332	**98,62**
Vote	Maksimum	97,727	**100**	**100**	**100**	**100**
	Standart sapma	3,122	4,921	4,205	3,424	**1,62**
	Ort. kural sayısı	9,8	29,1	17,8	**5,8**	18,8

Tablo 7.83.'de yer alan minimum ve ortalama tahminleyici doğruluklar incelendiğinde en yüksek değeri geliştirilen TAKKOYSA-sınıflandırıcısının elde ettiği görülmektedir. Maksimum tahminleyici doğruluğu ise NBTree algoritması hariç diğer dört algoritma da yakalayabilmiştir. Ancak 10 katta elde edilen sonuçlar arasında en fazla tutarlılık 1,62'lik bir standart sapma ile geliştirilen algoritmaya aittir.

Çıkarılan ortalama kural sayısı dikkate alındığında ise geliştirilen algoritma diğer algoritmalarla rekabet edebilecek düzeydedir. DecisionTable algoritmasına göre çok daha az kuralı daha yüksek ortalama doğruluk ile elde etmiştir. Diğer algoritmalardan daha yüksek kural sayısı içerse de elde edilen doğruluk değerleri daha yüksektir.

Algoritmalar ile elde edilen sonuçlar aralarında anlamlı bir farklılık olup olmadığının değerlendirilmesi için gerçekleştirilen Mann Whitney U testi sonuçları Tablo 7.84.'de gösterilmektedir. Mann Whitney U testi ile, algoritma medyan değerleri arasındaki farkın anlamlı olup olmadığı test edilmiştir. Tabloda algoritmaların medyan ve p değerleri gösterilmektedir.

Tablo 7.84.'deki test doğruluğu-medyan değerleri incelendiğinde geliştirilen algoritmanın medyan değerinin diğer dört algoritmanın medyan değerinden de büyük olduğu görülür. Ancak medyanlar arasındaki farkın istatistiksel olarak anlamlı olup olmadığını söylemek için p değerlerini incelemek gerekir. Test doğruluğu p değerleri incelendiğinde DecisionTable algoritması dışında kalan diğer üç algoritma ile geliştirilen algoritmanın elde ettiği tahminleyici doğruluklar arasında istatistiksel olarak anlamlı bir farklılık görülmektedir ($p<0,05$). Bu farklılık geliştirilen algoritmanın daha yüksek doğruluk elde edebildiğini ifade etmektedir. DecisionTable algoritması ile

geliştirilen algoritma test doğrulukları arasında istatistiksel olarak anlamlı bir farklılıktan söz edilemez çünkü p değeri önemlilik düzeyinden büyüktür (p>0,05).

Tablo 7.84. Vote veri kümesi için Mann Whitney U testi sonuçları.

	Etkinlik Ölçütü			
	Test Doğruluğu		Kural Sayısı	
Algoritma	Medyan	P değeri	Medyan	P değeri
TAKKOYSA-sınıflandırıcısı NBTree	98,865 94,266	0,0013	18 9,5	(-)0,0008
TAKKOYSA-sınıflandırıcısı Decision Table	98,865 96,565	**0,0963**	18 21	**0,0757**
TAKKOYSA-sınıflandırıcısı PART	98,865 93,182	0,0140	18 7	(-)0,0002
TAKKOYSA-sınıflandırıcısı C4.5	98,865 97,674	0,0452	18 6	(-)0,0002

Çıkarılan kural sayıları dikkate alındığında, karşılaştırmada kullanılan C4.5, PART ve NBTree algoritmalarının medyan değerlerinin TAKKOYSA-sınıflandırıcısından küçük olduğu görülür. p değerleri incelendiğinde DecisionTable algoritması dışında kalan üç algoritmanın istatistiksel olarak geliştirilen algoritmadan daha az sayıda kural ürettiği söylenebilir (p<0,05). DecisionTable ile geliştirilen algoritmanın bulduğu kural sayıları arasında istatistiksel olarak anlamlı bir farklılık yoktur (p>0,05).

Tablo 7.85. geliştirilen algoritma ile günümüz literatüründe yer alan kural tabanlı sınıflandırıcıların elde ettikleri tahminleyici doğruluklar ve ortalama kural sayılarının karşılaştırılmasını vermektedir.

Tablo 7.85.'de verilen değerler incelendiğinde, ortalama doğruluk etkinlik ölçütünde en iyi sonucu geliştirilen algoritma vermiştir. Kural sayıları dikkate alındığında ise geliştirilen algoritmanın en az sayıda kuralla kural kümeleri oluşturduğu söylenebilir. Elde edilen kural sayısı diğer algoritmalara göre oldukça düşüktür. Geliştirilen algoritmanın performansı bu veri kümesinde de oldukça yüksektir.

Tablo 7.85. Vote veri kümesinde TAKKOYSA-sınıflandırıcısının kural-tabanlı sınıflandırıcılar ile karşılaştırma sonuçları.

Veri Kümesi	**Algoritma**	**Referans**	**Kural Sayısı**	**Ortalama Doğruluk**
Vote	GARC	Chen et al. (2006)	32	89,67
	RMR	Thabtah ve Cowling (2007)	84	88,7
	***BP+GA	Tsukimoto ve Hatano (2003)	-	96,3
	TAKKOYSA-sınıflandırıcısı		**18,8**	**98,62**

Algoritma, diğer bir yapay sinir ağlarından kural çıkaran algoritma olan BP+GA ile karşılaştırıldığında ise daha yüksek test doğruluğu elde ettiği görülmektedir.

7.6.11. WBC Veri Kümesi İçin Karşılaştırma Sonuçları

Tablo 7.86. Wisconsin breast cancer veri kümesinde geliştirilen yapay sinir ağlarından kural çıkarım algoritmasına 10-katlı çapraz geçerlilik testi uygulanması ile elde edilen sonuçların NBTree, PART, DecisionTable ve C4.5 algoritmaları ile karşılaştırılmasını vermektedir. Karşılaştırmalar minimum, ortalama ve maksimum test doğruluğu, test doğruluğu standart sapması ve ortalama kural sayıları dikkate alınarak gerçekleştirilmiştir.

Tablo 7.86. WBC veri kümesi klasik sınıflandırıcılar ile karşılaştırma sonuçları.

Veri kümesi		NBTree	DT	PART	C4.5	Geliştirilen Algoritma
	Minimum	91,43	91,43	90	88,57	**97,14**
	Ortalama	96,28	95,42	93,85	94,56	**99**
WBC	Maksimum	**100**	**100**	98,57	98,57	**100**
	Standart sapma	2,36	2,68	2,94	3,63	**0,97**
	Ort. kural sayısı	**3,7**	44,7	9,4	12,2	24,1

Tablo 7.86.'da yer alan minimum, ortalama ve maksimum tahminleyici doğruluk değerleri incelendiğinde en iyi sonucu geliştirilen kural çıkarım algoritmasının verdiği

görülmektedir. Elde edilen sonuçlar arasındaki standart sapma ise 0,97 gibi çok küçük bir değerdir. Bu da kural çıkarım algoritmasının kaliteli çözümler üretebildiğini göstermektedir.

Çıkarılan ortalama kural sayısı bakımından ise geliştirilen algoritma 24,1 kural ile ortalama bir değer elde etmiştir. TAKKOYSA-sınıflandırıcısı, ikili veri kümeleri üzerinde çalışmasına rağmen elde ettiği kural sayıları, tahminleyici doğrulukla birlikte dikkate alındığında oldukça iyidir.

Elde edilen sonuçların daha iyi değerlendirilebilmesi ve aralarında anlamlı bir farklılık olup olmadığının analiz edilmesi için gerçekleştirilen Mann Whitney U testi sonuçları Tablo 7.87.'de gösterilmektedir. Algoritmaların medyan ve p değerleri tabloda yer almaktadır.

Tablo 7.87. WBC veri kümesi için Mann Whitney U testi sonuçları.

	Etkinlik Ölçütü			
	Test Doğruluğu		Kural Sayısı	
Algoritma	Medyan	P değeri	Medyan	P değeri
TAKKOYSA-sınıflandırıcısı NBTree	98,57 96,429	0,0140	24 3,5	(-)0,0002
TAKKOYSA-sınıflandırıcısı Decision Table	98,57 95,683	0,0036	24 41	**0,0588**
TAKKOYSA-sınıflandırıcısı PART	98,57 93,571	0,0013	24 9	(-)0,0002
TAKKOYSA-sınıflandırıcısı C4.5	98,57 95,714	0,0173	24 12,5	(-)0,0002

Tablo 7.87.'deki test doğruluğu-medyan değerleri incelendiğinde geliştirilen algoritmanın medyan değerinin diğer dört algoritmanın medyan değerinden büyük olduğu görülür. Test doğruluğu p değerleri incelendiğinde, geliştirilen algoritmanın elde ettiği tahminleyici doğruluklar arasında istatistiksel olarak anlamlı bir farklılık görülmektedir ($p<0,05$). Geliştirilen algoritmanın medyan değerinin de daha büyük

olması geliştirilen algoritmanın diğer algoritmalardan istatistiksel olarak daha iyi sonuçlar üretebildiğini göstermektedir.

Tablo 7.87.'de (-) işaretli değerler, karşılaştırılan algoritmanın geliştirilen TAKKOYSA-sınıflandırıcısından daha iyi sonuç verdiğini ifade etmektedir. Çıkarılan kural sayıları dikkate alındığında, karşılaştırmada kullanılan DecisionTable dışında kalan üç algoritmanın medyan değerlerinin geliştirilen algoritmadan küçük olduğu görülür. p değerleri incelendiğinde bu üç algoritmanın istatistiksel olarak geliştirilen algoritmadan daha az sayıda kural ürettiği söylenebilir ($p<0,05$).

Kural sayısı etkinlik ölçütünde DecisionTable algoritması ile geliştirilen algoritmanın elde ettiği kural sayıları arasında istatistiksel olarak bir farklılık yoktur, çünkü p değeri önemlilik düzeyinden büyüktür ($p>0,005$).

Tablo 7.88. geliştirilen algoritma ile günümüz literatüründe yer alan kural tabanlı sınıflandırıcıların elde ettikleri tahminleyici doğruluklar ve ortalama kural sayılarının karşılaştırmasını vermektedir. Tabloda, kural sayılarında bazı algoritmalara ait bilgi yer almamaktadır. Bunun nedeni ilgili algoritmanın ya her sınıf için sadece bir kural çıkarıyor olması veya kural sayısı bilgisinin bulunmamasıdır. *** ile gösterilen sınıflandırıcılar ise yapay sinir ağlarından kural çıkarım algoritmalarını ifade etmektedir.

Tablo 7.88. incelendiğinde ortalama doğruluk etkinlik ölçütünde en iyi sonucu % 99,41 doğrulukla MEPAR-miner algoritmasının verdiği görülmektedir. Geliştirilen algoritmanın ise ortalama test doğruluğu % 99'dur. Doğruluklar arasında önemli bir fark gözükmese de MEPAR-miner algoritmasının her sınıf için bir kural çıkararak bu doğruluğu elde etmesi geliştirilen algoritmaya üstünlük sağlamaktadır. Kalan diğer algoritmalarla karşılaştırıldığında ise geliştirilen algoritma daha yüksek doğruluk değeri elde etmiştir. Çıkarılan kural sayısı ise diğer algoritmalara kıyasla ortalama bir değerle makul seviyededir.

Tablo 7.88. WBC veri kümesinde TAKKOYSA-sınıflandırıcısının kural-tabanlı sınıflandırıcılar ile karşılaştırma sonuçları.

Veri Kümesi	**Algoritma**	**Referans**	**Kural Sayısı**	**Ortalama Doğruluk**
WBC	EvoC	Tan et al. (2003)	5.99	97,57
	MEPAR-miner	Baykasoğlu ve Özbakır (2007)	-	**99,41**
	-	Pitangui ve Zaverucha (2006)	**2**	96,14
	GP	Bojarczuk et al. (2004)	-	93,5
	MOGGP	Pappa ve Freitas (2008)	-	92,1
	Ant-Miner	Parpinelli et al. (2002b)	6,2	96,04
	ACO-Miner	Wang ve Feng (2004)	4,63	97,15
	Ant-Miner3	Liu et al. (2003)	13,2	94,32
	Unordered Rule set Ant-Miner	Smaldon ve Freitas (2006)	6,2	95,61
	E-Ant-Miner	Ji et al. (2006)	5,8	95,62
	AntMiner+	Martens et al. (2007)	3,4	96,4
	Ant-Miner with Imp. Quic k Red.	Jaganathan et al. (2007)	12,25	71,26
	-	Nalini ve Balasubramanie (2008)	7,97	97,26
	CPSO	Sousa et al. (2004)	7	93,4
	GARC	Chen et al. (2006)	21	94,85
	PSO/ACO2	Holden ve Freitas (2007)	-	93,85
	RMR	Thabtah ve Cowling (2007)	60	95,92
	TFPC	Coenen ve Leng (2007)	14	90
	CMAR	Coenen ve Leng (2007)	66,8	91,2
	CBA	Coenen ve Leng (2007)	13,7	94,1
	-	Wang ve Zhang (2005)	-	96,18
	TACO-miner	Thangavel ve Jaganathan, 2007	4,52	97,98
	***RullExt	Zhang vd. (1996)	9	96,64
	***G-REX oracle	Johansson vd., 2006	-	96,7
	***CGA	Hruschka ve Ebecken (2006)	3	96,35
	TAKKOYSA-sınıflandırıcısı		24,1	99

Algoritma, diğer yapay sinir ağlarından kural çıkaran algoritmalar ile karşılaştırıldığında ise en yüksek test doğruluğu değerini elde ettiği görülmektedir. Geliştirilen algoritmanın kural sayısı diğer iki algoritmadan yüksek çıkmıştır.

7.6.12. Zoo Veri Kümesi İçin Karşılaştırma Sonuçları

Zoo veri kümesi, az sayıda örnek içermesi nedeniyle tek bir eğitim ve test kümesine ayrılmıştı. Tablo 7.89. geliştirilen TAKKOYSA-sınıflandırıcısının Zoo veri kümesinde 10 kez aynı eğitim ve test veri kümelerinde çalıştırılması ile elde edilen sonuçların NBTree, PART, DecisionTable ve C4.5 algoritmalarıyla karşılaştırılmasını vermektedir. Karşılaştırmalar minimum, ortalama ve maksimum test doğruluğu, test doğruluğu standart sapması ve ortalama kural sayıları dikkate alınarak gerçekleştirilmiştir.

Tablo 7.89. Zoo veri kümesi klasik sınıflandırıcılar ile karşılaştırma sonuçları.

Veri kümesi		NBTree	DT	PART	C4.5	Geliştirilen Algoritma
	Minimum	90,625	84,848	87,879	87,879	**94,12**
	Ortalama	94,868	88,973	91,634	91,634	**95,3**
Zoo	Maksimum	**100**	93,939	96,875	96,875	**100**
	Standart sapma	3,542	3,25	2,696	2,696	**2,06**
	Ort. kural sayısı	**3,3**	14,1	7	7,1	8,9

Tablodan da görüldüğü gibi tahminleyici doğruluklar dikkate alındığında TAKKOYSA-sınıflandırıcısı karşılaştırmaya tabi tutulan dört algoritmadan daha iyi sonuçlar bulmuştur. Algoritmanın aynı eğitim ve test kümesinde 10 kez çalıştırılması sonucu elde edilen değerin standart sapması diğer algoritmalardan daha düşüktür. Bu da algoritmanın kaliteli çözümler ürettiğinin bir göstergesidir.

Çıkarılan ortalama kural sayısı bakımından geliştirilen algoritma 8,9 kural ile diğer algoritmalara yakın sayıda kural çıkarmıştır. En az sayıda kuralı NBTree algoritması, en çok kuralı ise DecisionTable algoritması üretmiştir. Geliştirilen algoritmanın elde ettiği ortalama kural sayısı, test doğruluğu ile birlikte değerlendirildiğinde oldukça iyidir.

Elde edilen sonuçların daha iyi değerlendirilebilmesi ve aralarında anlamlı bir farklılık olup olmadığının incelenmesi için gerçekleştirilen Mann Whitney U testi sonuçları

Tablo 7.90.'da gösterilmektedir. Tabloda algoritmaların medyan ve p değerleri gösterilmektedir.

Tablo 7.90. Zoo veri kümesi için Mann Whitney U testi sonuçları.

	Etkinlik Ölçütü			
	Test Doğruluğu		Kural Sayısı	
Algoritma	Medyan	P değeri	Medyan	P değeri
TAKKOYSA-sınıflandırıcısı NBTree	94,12 94,115	0,5708	9 3	(-)0,0002
TAKKOYSA-sınıflandırıcısı Decision Table	94,12 88,06	0,0002	9 14,5	0,0003
TAKKOYSA-sınıflandırıcısı PART	94,12 91,045	0,0073	9 -	-
TAKKOYSA-sınıflandırıcısı C4.5	94,12 91,045	0,0073	9 7	(-)0,0012

Tablo 7.90.'daki test doğruluğu-medyan değerleri incelendiğinde geliştirilen algoritmanın medyan değerinin diğer dört algoritmanın medyan değerinden büyük olduğu görülür. Ancak medyanlar arasındaki farkın istatistiksel olarak anlamlı olup olmadığını söylemek için p değerlerini incelemek gerekir. Eğer p-değeri α=0,05 önemlilik değerinden küçükse, karşılaştırılan algoritmaların medyanları arasında istatistiksel olarak anlamlı bir farklılık var demektir. Test doğruluğu p değerleri incelendiğinde geliştirilen algoritmanın elde ettiği tahminleyici doğruluklar arasında istatistiksel olarak anlamlı bir farklılık görülmektedir ($p<0,05$). Geliştirilen TAKKOYSA-sınıflandırıcısı, karşılaştırmada kullanılan dört algoritmadan istatistiksel olarak daha yüksek doğrulukta kural kümeleri üretebilmektedir.

Çıkarılan kural sayıları dikkate alındığında, PART algoritmasının 10 çalıştırmada da 7 kural bulması ve standart sapmasının 0 olması nedeniyle Mann Whitney U testi ile analiz yapılamamıştır. Ancak geliştirilen algoritmadan daha az sayıda kural çıkardığı söylenebilir.

NBTree ve C4.5 algoritmalarının kural sayısı medyan değerleri geliştirilen algoritmanın medyan değerinden küçüktür. p değerleri de önemlilik düzeyinden küçük olduğu için,

bu algoritmaların geliştirilen kural çıkarım algoritmasından istatistiksel olarak daha az sayıda kural kümeleri üretebildikleri söylenebilir.

Geliştirilen algoritma ile Decision Table algoritmasının kural sayısı medyan değerleri incelendiğinde geliştirilen algoritmanın daha az sayıda kural ürettiği görülür. P değeri ise bu durumun istatistiksel olarak anlamlı olduğunu ispatlamaktadır. Dolayısıyla geliştirilen algoritma DecisionTable algoritmasına göre az sayıda kural ile daha yüksek doğrulukta kurallar üretebilmektedir denilebilir.

Tablo 7.91. geliştirilen algoritma ile günümüz literatüründe yer alan kural tabanlı sınıflandırıcıların elde ettikleri tahminleyici doğruluklar ve ortalama kural sayılarının karşılaştırmasını vermektedir.

Tablo 7.91. Zoo veri kümesinde TAKKOYSA-sınıflandırıcısının kural-tabanlı sınıflandırıcılar ile karşılaştırma sonuçları.

Veri Kümesi	**Algoritma**	**Referans**	**Kural Sayısı**	**Ortalama Doğruluk**
Zoo	DOEA	Tan et al. (2006a)	7	92,1
	INPGA	Dehuri ve Mall (2006)	7	90,55
	CPSO	Sousa et al. (2004)	7	84,8
	PSO/ACO2	Holden ve Freitas (2007)	-	81,36
	RMR	Thabtah ve Cowling (2007)	8	95,15
	TFPC	Coenen ve Leng (2007)	314,9	93
	CMAR	Coenen ve Leng (2007)	43,6	94
	CBA	Coenen ve Leng (2007)	**2**	40,4
	GARC	Chen et al. (2006)	90	82,35
	***G-REX oracle	Johansson vd., 2006	-	95
	TAKKOYSA-sınıflandırıcısı		8,9	**95,3**

Tablo 7.91. incelendiğinde ortalama doğruluk etkinlik ölçütünde en iyi sonucu geliştirilen algoritmanın verdiği görülmektedir. Çıkarılan kural sayısı dikkate

alındığında ise geliştirilen algoritma, diğer algoritmalara göre ortalama bir kural sayısı ile en yüksek doğruluğu elde etmiştir.

Algoritma, yapay sinir ağları üzerinde çalışan bir diğer algoritma G-REX oracle ile karşılaştırıldığında yaklaşık aynı doğrulukları elde ettikleri görülmektedir.

7.7. Sonuç

Çalışmanın bu bölümünde geliştirilen TAKKOYSA-sınıflandırıcısının elde ettiği sonuçlar test doğruluğu ve ortalama kural sayısı etkinlik ölçütleri açısından değerlendirilmiştir. Değerlendirmede, literatürde mevcut olan 12 farklı ikili ve çok sınıflı referans veri kümesi kullanılmıştır. Her bir test problemine Taguchi yöntemi ile deney tasarımı uygulanarak, algoritma faktörleri performans ölçütü olan test doğruluğu üzerindeki etkilerine göre sıralanmıştır. Faktörlerin etkinlikleri analiz edilmiş ve en yüksek test doğruluğunu elde etmek için her bir faktörün alması gereken düzeyler belirlenmiştir.

- Faktörlerin veri kümesine göre önem sırası ve aldıkları düzeyler değişmektedir. Bunun nedeni veri kümelerindeki ikili değişken ve özellik sayılarının değişken olmasıdır. Mesela küçük veri kümelerinde karınca sayısı faktörü "M" etkisiz iken, veri kümesinin boyutu arttıkça etkin hale gelmektedir.

- Bütün veri kümelerinde gizli katman(lar)daki işlem elemanı sayısı faktörü "İES" etkin çıkmıştır. Veri kümelerindeki analizler sonucunda eğer İES faktörü, veri kümesine göre ikinci veya üçüncü düzeyinde kullanılırsa en yüksek test doğruluğu elde edilebilir sonucuna varılmıştır.

- Gizli katman sayısı "GK" bir diğer etkili faktördür. Bir kaç veri kümesi hariç bütün veri kümelerinde etkinlik sıralamasında ilk sırada yer almaktadır. Genel olarak veri kümelerinde gizli katman sayısı ilk değerinde iken yani bir gizli katman kullanılırsa test doğruluğu artar sonucuna varılmıştır.

- Çok katmanlı algılayıcılardaki iterasyon sayısı faktörü "Epoch", en etkisiz faktör olarak düşünülebilir. Çünkü 12 veri kümesinin beşinde etkisiz çıkmıştır.

Her bir veri kümesinde belirlenen faktör düzeyleri ile geliştirilen algoritmanın elde ettiği sonuçlar, ele alınan etkinlik ölçütleri bakımından incelenmiştir. Test doğruluğu etkinlik ölçütünde 12 veri kümesinde % 90-100 arasında ortalama doğruluk elde edilmiştir. Ortalama kural sayısı etkinlik ölçütünde ise veri kümelerinde ortalama 5 ila 50 kural çıkarılmıştır.

Bölümün sonunda algoritmanın performansı iki aşamada değerlendirilmiştir. İlk aşamada geliştirilen algoritma dört farklı klasik makine öğrenme algoritması ile karşılaştırılmış ve istatistiksel analizler gerçekleştirilmiştir. TAKKOYSA-sınıflandırıcısı, 12 veri kümesinde de karşılaştırıldığı her dört makine öğrenme algoritmasından daha yüksek test doğrulukları elde etmiştir. Ortalama kural sayısı bakımından ise iki veri kümesinde daha az kural çıkarmıştır.

İkinci aşamada ise algoritmanın elde ettiği sonuçlar literatürde mevcut olan kural tabanlı sınıflandırıcılar ile karşılaştırılmıştır. Bu karşılaştırmalara yapay sinir ağlarından kural çıkaran algoritmalar da dahil edilmiştir. Karşılaştırma sonuçları geliştirilen TAKKOYSA-sınıflandırıcısının diğer algoritmalarla rekabet edebilecek düzeyde doğru ve etkin kurallar çıkarabildiğini göstermektedir.

8. BÖLÜM
SONUÇ VE TARTIŞMALAR

8.1. Çalışmanın Katkıları

Bu bölümde çalışmanın bilim dünyasına katkıları ve orjinalliği ortaya konmuştur. Çalışmanın her bölümünün sonunda, o bölümde önerilen ve/veya uygulanan yöntemlerin getirdiği yeniliklere dair değerlendirmelere yer verilmiştir. Bu bölümde bu değerlendirmeler bir arada ele alınmıştır.

Günümüzde bilgisayar sistemlerindeki hızlı artışla birlikte üretilen veri boyutu da hızla artmıştır. Bu büyük boyutlu veri ise klasik istatistik teknikler ile çözülemez hale gelmiştir. Oluşan bu büyük boyutlu veri ile başa çıkmakta artık veri madenciliği yöntemleri yaygın bir şekilde kullanılmaktadır.

Birçok veri madenciliği fonksiyonunun içinde, veritabanlarında gizli olan bilgiyi kullanıcıların anlayabileceği kural listeleri veya karar ağaçları formunda sunan "sınıflandırma" dikkat çekmektedir. Literatürde pek çok araştırmacı, büyük boyutlu veri kümelerinde gizli olan bilgiyi anlaşılır ve doğru kurallar şeklinde sunan etkin sınıflandırma algoritmaları geliştirmek için çalışmalar yapmaktadırlar.

Yapay sinir ağları, sınıflandırmada oldukça yüksek doğruluklar elde etmekte, ancak elde ettikleri bilgileri kullanıcıların anlayabileceği dilsel kurallara dönüştürememektedirler. Bu nedenle son yıllarda eğilim, yapay sinir ağlarını farklı algoritmalar ile bütünleştirerek YSA'ların yapısında gizli olan bilgiyi keşfetmek şeklindedir.

Bu çalışmada da yapay sinir ağlarının elde ettikleri yüksek doğrulukları dilsel ifadelere dönüştürmek üzerine bir çalışma yapılmıştır. Geliştirilen TAKKOYSA-sınıflandırıcısı iki ana bölümden oluşmaktadır. İlk bölümde YSA modeli olarak çok katmanlı algılayıcılar eğitilmekte ve en az hatayı veren ağırlık grupları elde edilmektedir. İkinci aşamada ise bu en küçük hatayı veren ağırlık gruplarını ifade eden dilsel kurallar üretil-

mektedir. Çıkarılan ağırlık grupları kullanılarak YSA'ların çıktı fonksiyonlarının ve ağırlık gruplarını yansıtan kural kümelerinin en iyilenmesinde karınca koloni optimizasyonu algoritmasından faydalanılmıştır. KKO algoritmasının tercih edilmesinin nedeni, yapay sinir ağının çıktı fonksiyonunun en iyilenmesi probleminin bir optimizasyon problemi olması, KKO'nun uygulandığı alanlarda elde ettiği kaliteli sonuçlar ve literatürde bu alanda bir KKO uygulamasına rastlanmamış olmasıdır.

TAKKOYSA-sınıflandırıcısı yapay sinir ağlarının ve karınca koloni optimizasyonu algoritmasının avantajlarını birleştirmekte, böylece yüksek kalitede çözümler üretebilmektedir. Algoritmanın performansını değerlendirmek için literatürde mevcut olan 12 ikili ve çok-sınıflı referans veri kümesinden faydalanılmıştır. Deney tasarımı ile her bir veri kümesinde algoritmanın etkin parametreleri ve düzeyleri belirlenmiştir. Veri kümeleri üzerinde yapılan analizler algoritmanın oldukça doğru ve özlü kurallar çıkardığını göstermiştir.

Ayrıca algoritmanın elde ettiği sonuçlar çeşitli kural tabanlı sınıflandırıcılar ile karşılaştırılmıştır. C4.5, PART gibi klasik makine öğrenme algoritmaları ile karşılaştırmalar, geliştirilen algoritmanın baskın bir şekilde diğer algoritmalardan iyi sonuçlar ürettiğini göstermektedir. Modern sınıflandırıcılar ile karşılaştırmalarda algoritma doğruluk açısından bütün algoritmalardan üstündür ve kural sayıları çoğu algoritma ile rekabet edebilecek düzeydedir. Eğitilmiş yapay sinir ağlarından kural çıkarım algoritmaları ile karşılaştırmalar da algoritmanın etkin bir şekilde yüksek doğrulukta daha iyi kural kümeleri üretebildiğini göstermektedir.

Tez çalışmasında elde edilen sonuçlardan iki SCI makalesi gönderilmiştir [187, 188]. Makaleler inceleme aşamasındadır. Ayrıca bir kitap bölümü [4], bir uluslar arası kongre [189] ve bir ulusal kongre bildirisi [190] sunulmuştur.

8.2. İleriye Yönelik Öneriler

Yapay sinir ağlarının kara kutu olması ve elde ettikleri sonuçları açıklayamamaları bilim dünyasını bu problemi çözmek için yöntemler geliştirmeye zorlamıştır. Literatürde YSA'lardan kural çıkarımıyla ilgili çok çeşitli çalışmalar mevcuttur. Bu alana olan büyük ilgi ve geliştirilen TAKKOYSA-sınıflandırıcısının yüksek kalitede çözümler üretmesi nedeniyle ileriye yönelik bazı önerilerde bulunulabilir.

- Kural indirgemede tek bir uygunluk fonksiyonunda tek bir amaç yerine birden fazla amaç kullanılarak farklı etkinlik ölçütleri eşzamanlı olarak en iyilenebilir. Örnek olarak uygun çok amaçlı fonksiyon tasarlanarak test doğruluğu en büyüklenirken, kural sayısı en küçüklenebilir.

- Yapay sinir ağlarının eğitiminde ÇKA gibi klasik YSA modelleri yerine genetik algoritma, parça sürü optimizasyonu gibi meta-sezgisel algoritmalar kullanılabilir ve YSA'ların hataları en küçüklenebilir.

- Kural kümesinin tahminleyici doğruluğu en iyilenecek şekilde YSA topolojisi optimizasyonu gerçeklenebilir.

- YSA'ların eğitimi ve kural çıkarımı prosedürleri entegre edilerek YSA'nın her iterasyonunda bir kural kümesi üretilerek, bu kural kümesi iterasyonlar boyunca değişen ağırlıklardan faydalanılarak en iyilenebilir.

- Tur atan karınca koloni optimizasyonu algoritmasının erken yakınsamasını engellemek için tabu aramanın sıklık faktörü yerine farklı yöntemler incelenerek, algoritmanın performansı geliştirilebilir.

Bu tez çalışmasında, eğitilmiş yapay sinir ağlarından karınca koloni optimizasyonu ile sınıflandırma kural çıkarımı ele alınmıştır. Çalışmada YSA modeli olarak çok katmanlı algılayıcı modeli ve KKO olarak tur atan karınca koloni optimizasyonu algoritması kullanılmıştır. Geliştirilen algoritma ile elde edilen sonuçların, bilim dünyasında diğer çalışmalara ışık tutacak katkılar sağlaması ve literatürde YSA'dan kural çıkarımında KKO'nun kullanılması boşluğunu doldurması amaçlanmıştır.

KAYNAKLAR

1. Han, J., Kamber, M., Data Mining: Concepts and Techniques, Morgan Kaufmann Publishers, San Fransisco, 2001.
2. Parpinelli, R. S., Lopes, H. S., Freitas, A. A., An Ant Colony Algorithm for Classification Rule Discovery, in Data Mining: A Heuristic Approach, London: Idea Group Publishing, Part IV, Chapter X, 191-208, 2002.
3. Tan, C., Yu, Q., Ang, J. H., A Dual-objective Evolutionary Algorithm for Rules Extraction in Data Mining, Computational Optimization and Applications, 34, 273-294, 2006.
4. Özbakır, L., Baykasoğlu, A., Kulluk, S., Rule Extraction from Neural Networks Via Ant Colony Algorithm for Data Mining Applications, in V. Maniezzo, R. Battiti, J. –P. Watson (Eds): Proceedings of the 2nd International Conference on Learning and Intelligent Optimization -LION 2007, Springer, Lecture Notes in Computer Science 5313, 177–191, 2008.
5. Kuttiyil, A. S., Survey of Rule Extraction Methods, Yüksek Lisans Tezi, Wayne State Üniversitesi, 2004.
6. Bologna, G., Is The Worth Generating Rules From Neural Network Ensembles?, Journal of Applied Logic, 2, 325-348, 2004.
7. Hruschka, E. R., Ebecken, N. F. F., Extracting Rules from Multilayer Perceptrons in Classification Problems: A Clustering-based Approach, Neurocomputing, 70, 384-397, 2006.
8. Andrews, R., Diederich, J, Tickle, A. B, A Survey and Critique of Techniques for Extracting Rules From Trained Artificial Neural Networks, Knowledge-based Systems, 8(6), 373-389, 1995.
9. Gallant, S. I., Connection Expert Systems, Communications of the ACM, 31 (2), 152-169, 1988.
10. Dorigo, M., Optimization, Learning and Natural Algorithms, PhD Thesis, Politecnico di Milano, Italy, 1992.
11. Parpinelli, R. S., Lopes, H. S., Freitas, A. A., Data Mining with An Ant Colony Optimization Algorithm, IEEE Trans on Evolutionary Computation, Special Issue on Ant Colony Algorithms, 6(4), 321-332, 2002.

12. Hiroyasu, T., Miki, M., Ono, Y., Minami, Y., Ant Colony for Continuous Functions, The Science and Engineering, Doshisha University, XX(Y), 2000.
13. Piatetsky-Shapiro, G., Knowledge Discovery in Real Databases: A Workshop Report, AI Magazine, 11(5), 68-70, 1991.
14. Fayyad, U. G., Piatetsky-Shapiro, P., Smyth, R., Uthurusamy, R., Advances in Knowledge Discovery and Data Mining, MA:MIT Pres, Cambridge, 1996.
15. Fayyad, U., Piatetsky-Shapiro, G., Smyth, P., The KDD Process for Extracting Useful Knowledge from Volumes of Data, Communications of ACM, 39(11), 27-34, 1996.
16. Frawley, W. J., Piatetsky-Shapiro, G., Matheus, C. J., Knowledge Discovery Databases: An Overview, p.1-27, In Knowledge Discovery In Databases (G. Piatetsky-Shapiro and W. J., Frawley eds.), MA:AAAI/MIT, Cambridge, 1991.
17. Raghavan, V. V., Sever, H., The State of Rough Sets for Database Mining Applications, T. Y. Lin (ed.), 23rd Computer Science Conference Workshop on Rough Sets and Database Mining, p. 1-11, San Jose, CA, 1994.
18. Swift, R., Accelerating Customer Relationship, Prentice Hall PTR, New Jersey, 2001.
19. Sever H., Oğuz, B., Veritabanlarında Bilgi Keşfine Formal bir Yaklaşım, Kısım 1: Eşleştirme Sorguları ve Algoritmalar, Bilgi Dünyası, 3(2), 173-204, 2002.
20. Quinlan, J. R., Introduction of Decision Trees, Machine Learning, I, 81-106, 1986.
21. Chan, K. C. C., Wong, A. K. C., A Statistical Technique for Extracting Classificatory Knowledge from Databases, In Knowledge Discovery in Databases (G. Piatetsky-Shapiro and W. J. Frawley, eds.), 107-123, Cambridge, MA:AAA/MIT, 1991.
22. Lee, S. K., An Extended Relational Database Model for Uncertain and Imprecise Information, In Proceedings of The 1st VLDB Conference, Vancouver, British Columbia, Canada, 211-218, 1992.
23. Luba, T., Lasocki, R., On Unknown Attribute Values in Functional Dependicies, In Proceedings of The International Workshop on Rough Sets and Soft Computing, San Jose, CA, 490-497, 1994.
24. Grzymala-Busse, J. W., On The Unknown Attribute Values in Learning from Examples, In Proceedings of Methodologies for Intelligent Systems (Z. W. Ras

and M. Zeinankowa, eds.), Lecture Notes in Artificial Intelligence, New York: Springer-Verlag, 542, 368-377, 1991.

25. Thiesson, B., Accelerated Quantification of Bayesian Networks with Incomplete Data, In the 1st International Conference on Knowledge Discovery and Data Mining (U. Fayyad and R. Uthurusamy, eds.), Montreal, Quebec, Canada, 306-311, 1995.
26. Aggarwal, C. C., Parthasarathy, S., Mining Massively Incomplete Data Sets by Conceptual Reconstruction, International Conference on Knowledge Discovery and Data Mining, Proceedings of The 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Fransisco, California, 227-232, 2001.
27. Peng, H., Zhu, S., Handling of Incomplete Data Sets Using ICA and SOM in Data Mining, Neural Computing & Applications, 16(2), 167-172, Springer, 2007.
28. Deogun, J. S., Raghavan, V. V., Sever, H., Exploiting Upper Approximations in The Rough Set Methodology, In the 1st International Confernece on Knowledge Discovery and Data Mining (U. Fayyad and R. Uthurusamy, eds.), Montreal, Quebec, Canada, 69-74, 1995.
29. Aydoğan, F., E-ticarette Veri Madenciliği Yaklaşımlarıyla Müşteriye Hizmet Sunan Akıllı Modüllerin Tasarımı ve Gerçekleştirimi, Hacettepe Üniversitesi, Fen Bilimleri Enstitüsü, Bilgisayar Mühendisliği Anabilim Dalı, Yüksek Lisans Tezi, 2003.
30. Adriaans, P., Zantige, D., Data Mining, Harlow: Addison-Wesley, 158, 1996.
31. Chen Z., Data Mining and Uncertain Reasoning, A Willey-Interscience Publication, USA, 392, 2001.
32. Weiss, S. M., Kulikowski, C. A., Computer Systems That Learn: Classification and Prediction Methods from Statistics, Neural Nets, Machine Learning and Expert Systems, Morgan Kaufman, 1991.
33. Ye, N., The Handbook of Data Mining, Lawrance Erlbaum, 1st edition, 2003.
34. Michalski, R. S., Step, R. E., Learning from Observation: Conceptual Clustering. In R. S. Michalski, J. G. Oneli C., and Mite T. M., hell. Editors, Machine Learning: An Artificial Intelligence Approach, 1, Morgan Kaufmann, 331-363, 1983.

35. Jain, K. K., Dubes, R. C., Algorithms for Clustering Data, Englewood Cliffs, NJ: Prentice Hall, 1988.
36. Mitra, S., Acharya, T., Data Mining: Multimedia, Soft Computing and Bioinformatics, Wiley-Interscience, Inc. Hoboken, New Jersey, 2003.
37. Agrawal, R., Imielinski, T., Swami, A., Mining Association Rules Between Sets of Items in Large Databases, In ACM SIGMOD Conf. Managemenet of Data, 1993.
38. Bock, H., The Goal of Classification, Chapter 16.1.1, In Ed.: Klösgen W., Zytkow J. M., Handbook of Data Mining and Knowledge Discovery, Oxford University Pres, 2002.
39. Zhang, C., Zhang, S., Association Rule Mining: Models and Algorithms, Springer-Verlag Berlin Heidelberg, 2002.
40. Murthy, S. K., Automatic Construction of Decision Trees from Data: A Multi-disciplinary Survery, Data Mining and Knowledge Discovery, 2, 345-389, 1998.
41. Quinlan, R., C4.5: Programs for Machine Learning, Morgan Kaufmann Publishers, San Mateo: CA, 1993.
42. Haykin, S., Neural Networks: A Comprehensive Foundation. New York: Macmillan College Publishing Co., 1994.
43. Hush, D. R., Horne, B. G., Progress in Supervised Neural Networks, IEEE Signal Processing Magazine, 8-39, 1993.
44. Setiono, R., Leow, W. K., Thong, J. Y-L., Opening The Neural Network Blackbox: An Algorithm for Extracting Rules From Function Approximating Neural Networks, In Proceedings of ICIS 2000, International Conference on Information Systems, Brisbane, Australia, 2000.
45. Michalewicz, Z., Genetic Algorithms + Data Structure = Evolutionary Programs, 2nd Edition, Springer-Verlag, Berlin, 1996.
46. Holland, J. H., Adaption in Natural and Artificial Systems, University of Michigan Pres, Ann Arbor, MI, 1975.
47. Koza, J. R., Genetic Programming: On The Programming of Computers by Means of Natural Selection, MIT Press, Cambridge, 1992.
48. Fogel, D. B., An Evolutionary Approach to The Travelling Salesman Problem, Biological Cybernetics, 60, 139-144, 1988.

49. Rechenberg, I., Evolutionstrategie (Evolution Strategy), Formmann-Holzboog, Stuttgart, 1973.
50. Wong, M. L., Leung, K. S., Data Mining Using Grammar Based Genetic Programming and Applications, Kluwer Academic Publishers, London, 1999.
51. Beck, M. S. L. G., Applied Regression An Introduction, London: Sage Publications, 1991.
52. Tou, J. T., Gonzalez, R. C., Pattern Recognition Principles, London: Addison-Wesley, 1974.
53. Witten, I. H., Frank, E., Data Mining: Practical Machine Learning Tools and Techniques, The Morgan Kaufmann Publishers, San Fransisco, CA, 2005.
54. Edelstein, H. A., Introduction to Data Mining and Knowledge Discovery, Third Edition, Two Crows Corporation, 1999.
55. Bonabeau, E., Theraulaz, G., Swarm Smarts, Scientific American Inc., 72-79, 2000.
56. Beni, G., Wang, J., Swarm Intelligence in Cellular Robotic Systems, Proceed. NATO Advanced Workshop on Robots and Biological Systems, Tuscany, Italy, 1989.
57. Kennedy, J., Eberhart, R., Particle Swarm Optimization, In Proc. of The IEEE Int. Conf. on Neural Networks, Piscataway, NJ, 1942-1948, 1995.
58. Bishop, J. M., Stochastic Searching Networks, Proc. 1st IEEE Conf. on Artificial Neural Networks, 329-331, 1989.
59. Schank, R., Dynamic Memory: A Theory of Learning in Computers and People, New York: Cambridge University Press, 1982.
60. Kolodner, J. L., Case-Based Reasoning, San Mateo: Morgan Kaufmann, 1993.
61. Aha, D. W., Kibler, D., Albert, M. K., Instance-based Learning Algorithms, Machine Learning, 6, 37-66, 1991.
62. Pawlak, Z., Rough Sets Theoretical Aspects of Reasoning About Data, Kluwer Academic Publishers, 1991.
63. Binay, H. S., Yatırım Kararlarında Kaba Küme Yaklaşımı, Doktora Tezi, 2002.
64. Pawlak, Z., Skowron, A., Rough Set Rudiments, The International Workshop on Rough Sets and Soft Computing, San Jose, California, 72, 1994.
65. Zadeh, L. A., Fuzzy Sets, Information and Control, 8(3), 338-353, 1965.

66. Tan, K. C., Yu, Q., Heng, C., M., Lee, T. H., Evolutionary Computing for Knowledge Discovery in Medical Diagnosis, Artificial Intelligence in Medicine, 27, 129-154, 2003.
67. Nel, G., A Memetic Genetic Program for Knowledge Discovery, Master of Science Thesis, University of Pretoria, Pretoria, 2004.
68. Breiman, L., Friedman, J., Olshen, R., Stone, C., Classification and Regression Trees, Belmont: Wadsworth, 1984.
69. Falco, I. De, Cloppa, A., Della, Tarantino, E., Discovering Interesting Classification Rules with Genetic Programming, Applied Soft Computing, Elsevier Science Publishing, Amsterdam, 257-269, 2002.
70. Rouwhorst, S. E., Engelbrecht, A. P., Searching The Forest: A Building Block Approach to Genetic Programming for Classification Problems in Data Mining, Master of Thesis, Department of Mathematics and Informatics, Vrije Universiteit Amsterdam, The Netherlands, Research performed at University of Pretoria, South Africa, 2000.
71. Tan, C., Yu, Q., Ang, J. H., A Coevolutionary Algorithm for Rules Discovery in Data Mining, International Journal of System Science, 37(12), 835-864, 2006.
72. Uran, G., A Hybrid Heuristic Model for Classification Rule Discovery, Submitted in Partial Fulfillment of The Requirements for The Degree of Doctor of Professional Studies in Computing at School of Computer Science and Information Systems, Pace University, 2005.
73. Larose, D. T., Discovering Knowledge in Data: An Introduction to Data Mining, Wiley Interscience, New Jersey, 2005.
74. Kaur, H., Wasan, S. K., Al-Hegami, A. S., Bhatnagar, V., A Unified Approach for Discovery of Interesting Association Rules in Medical Databases, P. Perner (Ed.): ICDM 2006, LNAI 4065, Springer-Verlag Berling Heidelberg, 53-63, 2006.
75. Yao, Y., Zhou, B., Micro and Macro Evaluation of Classification Rules, Proc. 7th IEEE Int. Conf. on Cognitive Informatics (ICCI'08), Y. Wang, D. Zhang, J. –C. Latombe, and W. Kinsner (Eds.), 2008.
76. Bojarczuk, C. C., Lopes, H. S., Freitas, A. A., Michalkiewicz, E. L., A Constrained-Syntax Genetic Programming System for Discovering Classification Rules: Application to Medical Data Sets, Artificial Intelligence in Medicine, 30 (1), 27-48, 2004.

77. Spears, W. M., De Jong, K. A., Using Genetic Algorithms for Supervised Concept Learning, Machine Learning, 13 (2-3), 1993.
78. Noda, E., Freitas, A. A., Lopes, H. S., Discovering Interesting Rules with A Genetic Algorithm, Proc. Congress on Evolutionary Computation (CEC-99), 1322-1329, Washington D. C., USA, 1999.
79. Freitas, A. A., A Survey of Evolutionary Algorithms for Data Mining and Knowledge Discovery in Advances in Evolutionary Computation, Springer-Verlag, 2002.
80. Dharl, V., Chou, D., Provost, F., Discovering Interesting Patterns for Investment Decision Making with GLOWER L – A Genetic Learning Overlaid With Entropy Reduction, Journal of Data Mining and Knowledge Discovery, 2000.
81. Carvalho, D. R., Freitas, A. A., A Genetic-Algorithm for Discovering Small-disjunct Rules in Data Mining, Applied Soft Computing, 75-88, 2002.
82. Hsu, W. W., Hsu, C.-C., GEC: An Evolutionary Approach for Evolving Classifiers, M.-S. Chen, P. S. Yu, and B. Liu (Eds.):PAKDD 2002, LNAI 2336, 450-455, Springer-Verlag Berlin Heidelberg, 2002.
83. Pappa, G. L., Freitas, A. A., Towards A Genetic Programming Algorithm for Automatically Evolving Rule Induction Algorithms, In: ECML/PKDD 2004-Workshop on Advances in Inductive Learning, Pisa, 93-108, 2004.
84. Carvalho, D. R., Freitas, A. A., A Hybrid Decision Tree/Genetic Algorithm Method for Data Mining, Information Sciences, 163, 13-35, 2004.
85. Wang, Z., Zhang, D., Immunity-Based Genetic Algorithm for Classification Rule Discovery, L. Wang, K. Chen, and Y. S. Ong (Eds.): ICNC 2005, LNCS, 3611, 727-734, Springer-Verlag Berlin Heidelberg, 2005.
86. Tan, K. C., Yu, Q., Lee, T. H., A Distributed Evolutionary Classifier for Knowledge Discovery in Data Mining, IEEE Transactions on Systems, Man and Cybernetics, Part C: Applications and Reviews, 2005.
87. Chen, T.-C., Hsu, T.-C., A Gas Based Approach for Mining Breast Cancer Pattern, Expert Systems with Applications, 30, 674-681, 2006.
88. Dehuri, S., Mall, R., Predictive and Comprehensible Rule Discovery Using A Multi-Objective Genetic Algorithm, Knowledge-Based Systems, 19, 413-421, 2006.

89. Pitangui, C., Zaverucha, G., Genetic Based Machine Learning: Merging Pittsburgh and Michigan, An Implicit Feature Selection Mechanism and A New Crossover Operator, Proceedings of the 6th International Conference on Hybrid Intelligent Systems (HIS'06), 2006.
90. Baykasoğlu, A., Özbakır, L., MEPAR-miner: Multi-expression Programming for Classification Rule Mining, European Journal of Operational Research, 183, 767-784, 2007.
91. Patterson, G., Zhang, M., Fittness Functions in Genetic Programming for Classification with Unbalanced Data, M. A. Orgun and J. Thornton (Eds.): AI 2007, LNAI 4830, 769-775, 2007.
92. Dehuri, S., Patnaik, S., Ghosh, A., Mall, R., Application of Elitist Multi-Objective Genetic Algorithm for Classification Rule Generation, Applied Soft Computing, 8, 477-487, 2008.
93. Pappa, G. L., Freitas, A. A., Evolving Rule Induction Algorithms with Multi-Objective Grammer-Based Genetic Programming, Knowl Inf Syst, DOI 10.1007/s10115-008-0171-1.
94. Liu, B., Abbass, H. A., McKay, B., Classification Rule Discovery with Ant Colony Optimization, Proceedings of the IEEE/WIC International Conference on Intelligent Agent Technology (IAT'03), 2003.
95. Sousa, T. Silva, A., Neves, A., Particle Swarm Based Data Mining Algorithms for Classification Tasks, Parallel Computing, 30, 767-783, 2004.
96. Wang, Z., Feng, B., Classification Rule Mining with An Improved Ant Colony Algorithm, G. I. Webb and X. Yu (Eds.): AI 2004, LNAI 3339, Springer-Verlag Berling Heidelberg, 357-367, 2004.
97. Holden, N., Freitas, A. A., A Hybrid Particle Swarm/Ant Colony Algorithm for the Classification of Hierarchical Biological Data. In: Proc. 2005 IEEE Swarm Intelligence Symposium (SIS-05), 100-107, 2005.
98. Smaldon, J., Freitas, A. A., A New Version of the Ant-Miner Algorithm Discovering Unordered Rule Sets, In Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2006), 43-50, 2006.
99. Admane, L., Benatchba, K., Koudil, M., Siad, L., Maziz, S., AntPart: An Algorithm for the Unsupervised Classification Problem Using Ants, Applied Mathematics and Computation, 180, 16-28, 2006.

100. Ji, J., Zhang, N., Liu, C., An Ant Colony Optimization Algorithm for Learning Classification Rules, Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence (WI 2006 Main Conference Proceedings), 2006.
101. De Falco, I., Della Cioppa, A., Tarantino, E., Facing Classification Problems with Particle Swarm Optimization, Applied Soft Computing, 7(3), 652-658, 2007.
102. Martens, D., De Backer, M., Haesen, R., Vanthienen, J., Snoeck, M., Baesens, B., Classification with Ant Colony Optimization, IEEE Transactions on Evolutionary Computation, 11(5), 651-665, 2007.
103. Zahiri, S.-H., Seyedin, S.-A., Swarm Intelligence Based Classifiers, Journal of the Franklin Institute, 344, 362-376, 2007.
104. Holden, N., Freitas, A. A., A Hybrid PSO/ACO Algorithm for Classification, Genetic and Evolutionary Computation Conference, Proceedings of the 2007 GECCO Conference Companion on Genetic And Evolutionary Computation, London, United Kingdom, 2745-2750, 2007.
105. Jaganathan, P., Thangavel, K., Pethalakshmi, A., Karnan, M., Classification Rule Discovery with Ant Colony Optimization and Improved Quick Reduct Algorithm, IAENG International Journal of Computer Science, 33(1), IJCS_33_1_9, 2007.
106. Thangavel, K., Jaganathan, P., Rule Mining Algorithm with a New Ant Colony Optimization Algorithm, International Conference on Computational Intelligence and Multimedia Applications, 2007.
107. Otero, F. E., Freitas, A. A., Johnson, C. G., cAnt-Miner: An Ant Colony Classification Algorithm to Cope with Continuous Attributes, Lecture Notes in Computer Science, 5217, Proceedings of the 6th International Conference on Ant Colony Optimization And Swarm Intelligence, Brussels, Belgium, Springer-Verlag Berlin, Heidelberg, 48-59, 2008.
108. Nalini, C., Balasubramanie, P., Discovering Unordered Rule Sets for Mixed Variables Using Ant Colony Optimization, Journal of Computational Intelligence in Bioinformatics, 1(2-3), 129-140, 2008.
109. Otero, F. E. B., Freitas, A. A., Johnson, C. G., Handling Continuous Attributes in Ant Colony Classification Algorithms, To Appear in Proc. 2009 IEEE Symposium on Computational Intelligence in Data Mining, 2009.

110. Chen, G., Liu, H., Yu, L., Wei, Q., Zhang, X., A New Approach to Classification Based on Association Rule Mining, Decision Support Systems, 42, 674-689, 2006.
111. Thabtah, F. A., Cowling, P. I., A Greedy Classification Algorithm Based on Association Rule, Applied Soft Computing, 7, 1102-1111, 2007.
112. Coenen, F., Leng, P., The Effect of Threshold Values on Association Rule Based Classification Accuracy, Data & Knowledge Engineering, 60, 345-360, 2007.
113. Haykin, S., Neural Networks: A Comprehensive Foundation, New York: Macmillan College Publishing Co., 1994.
114. Hush, D. R., Horne, B. G., Progress in Supervised Neural Networks, IEEE Signal Processing Magazine, 8-39, 1993.
115. Öztemel, E., Yapay Sinir Ağları, İstanbul, Papatya Yayıncılık, 2003.
116. Anderson, D., McNeill, G., Artificial Neural Networks Technology, New York, Kaman Sciences Corporation, 1992.
117. Mehra, P., Wah, B. W., Artificial Neural Networks Concepts and Theory, IEEE Computer Society Press, Washington USA, 45, 1992.
118. Mitra, S., Acharya, T., Data Mining: Multimedia, Soft Computing, and Bioinformatics, Wiley-Interscience, 2003.
119. Sağıroğlu, Ş., Beşdok, E., Erler, M., Mühendislikte Yapay Zeka Uygulamaları-I: Yapay Sinir Ağları, UFUK Kitap Kırtasiye-Yayıncılık Tic. Ltd. Şti., 2003.
120. Elmas, Ç., Yapay Sinir Ağları, Ankara, Seçkin Yayıncılık, 2003.
121. Berry, M. J. A., Linoff, G. S., Data Mining Techniques: For Marketing, Sales, and Customer Relationship Management, Second Edition, Wiley Publishing, Inc., Indianapolis, Indiana, 2004.
122. Zurada, J. M, Artificial Neural Systems, PWS Publishing Company, St. Paul, MN, 1995.
123. Hebb, D. O., The Organization of Behavior, John Willey & Sons, New York, 1949.
124. Kohonen, T., The Self-organizing Map, Proc IEEE, 78 (9), 1990.

125. Curram, S. P., Minger, J., Neural Networks, Decision Tree Induction and Discriminant Analsis: Embrical Comparison, 1994.

126. Jacobs, R. A., Increased Rate of Convergence Through Learning Rate Adaptation, Neural Networks, 1, 295-307, 1988.

127. Fahlman, S. E., An Emprical Study of Learning Speed in Backpropagation Networks, Technical Report CMU-CS-88-162, Carnegie Mellon University, 1988.

128. Hagan, M. T. Demuth, H. B., Beale, M. H., Neural Network Design, Boston, MA: PWS Publishing, 1996.

129. Bishop, C., Neural Networks for Pattern Recognition, Oxford University Press., Oxford, 1995.

130. Rumelhart, D. E., Hinton, G. E., Williams, R. J., Learning Representations by Backpropogation Errors, Nature, 323, 533-536, 1986.

131. Kantardzic, M., Data Mining: Concepts, Models, Methods, and Algorithms, John Wiley & Sons, 2003.

132. Manic, M., Techniques in Neural Network Training with An Enhanced Robustness, Dissertation Presented in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy with a Major in Computer Science, Colloge of Graduate Studies, University of Idaho, 2003.

133. Rosenblatt, F., The Perceptron: A Probabilistic Model for Information Storage and Organization in The Brain, Psychological Review, 65, 386-408, 1958.

134. Craven, M. W., Shavlik, J. W., Using Sampling and Queries to Extract Rules from Trained Neural Networks, Machine Learning: Proceedings of the 11th International Conference, San Francisco CA, 1994.

135. Thrun, S. B., Extracting Provably Correct Rules from Artificial Neural Networks, Technical Report IAI-TR-93-5, Institut fur Informatik III Universtat Bonn, 1994.

136. Taha, I., Ghosh, J., Symbolic Interpretation of Artificial Neural Networks, IEEE Transactions on Knowledge and Data Engineering, 11(3), 448-463, 1999.

137. Zhang, Z., Zhou, Y., Lu, Y., Zhang, B., Extracting Rules from A GA-pruned Neural Network, Systems, IEEE International Conference on Man and Cybernetics, 3, 1682-1685, Oct. 14-17,1996.

138. Arbatlı, A. D., Akın, H. L., Rule Extraction from Trained Neural Networks Using Genetic Algorithms, Nonlinear Analysis Theory, Methods & Applications, 30 (3), 1639- 1648, 1997.

139. Fukumi, M., Akamatsu, N., Rule Extraction from Neural Networks Trained Using Evolutionary Algorithms with Deterministic Mutation, Proceedings IEEE World Congress on Computational Intelligence, IEEE International Joint Conference on Neural Networks, 1998.

140. Fukumi, M., Mitsukura, Y., Akamatsu, N., A New Rule Generation Method from Neural Networks Formed Using a Genetic Algorithm with Virus Infection, Proc. of IJCNN'2000,.44_03.1-6, Como, Italy, 2000.

141. Santos, R. T., Nievola, J., C., Freitas, A. A., Extracting Comprehensible Rules from Neural Networks via Genetic Algorithms, In Proc. 2000 IEEE Symp. On Combinations of Evolutionary Computation and Neural Networks (ECNN-2000), 130-139, San Antonio, TX, USA, 2000.

142. Hruschka, E. R., Ebecken, N. F. F., Applying A Clustering Genetic Algorithm for Extracting Rules From A Supervised Neural Network, In: IEEE International Joint Conference on Neural Networks (IJCNN 2000), Como, Italy, 2000.

143. Hruschka, E. R., Ebecken, N. F. F., Using A Clustering Genetic Algorithm for Rule Extraction From Artificial Neural Networks, IEEE Symposium on Combinations of Evolutionary Computation and Neural Network, 11 (13), 199-206, 2000.

144. Fu, X., Wang, L., Rule Extraction By Genetic Algorithms Based on A Simplified RBF Neural Network, Evolutionary Computation, Proceedings of the 2001 Congress, 2, 753-758, Seoul, South Korea, 2001.

145. Fukumi, M., Mitsukura, Y., Akamatsu, N., Knowledge Incorporation and Rule Extraction in Neural Networks, G. Dorffner, H. Bischof, and K. Hornik (Eds.): ICANN 2001, LNCS 2130, 1248-1253, Springer-Verlag Berling Heidelberg, 2001.

146. Dorado, J., Rabunal, J. R., Rivero, D., Santos, A. Pazos, A., Automatic Recurrent ANN Rule Extraction with Genetic Programming, In Proceedings of

The 2002 International Joint Conference on Neural Networks IJCNN'02, 1552-1557, 2002.

147. Tsukimoto, H., Hatano, H., The Functional Localization of Neural Networks Using Genetic Algorithms, Neural Networks, 16, 55-67, 2003.

148. Markowska-Kaczmar, U., Wnuk-Lipinski, P., Rule Extraction From Neural Network By Genetic Algorithm with Pareto Optimization, Artificial Intelligence and Soft Computing- ICAISC 2004, 7th International Conference, Proceedings, Springer, Lecture Notes in Computer Science, 3070, 450-455, Poland, 2004.

149. Elalfi, E., Haque, R., Elalami, M. E., Extracting Rules From Trained Neural Network Using GA for Managing E-business, Applied Soft Computing, 4, 65-77, 2004.

150. Markowska-Kaczmar, U., The Influence of Prameters in Evolutionary Based Rule Extraction Method From Neural Network, Proceedings of the 2005 5th International Conference on Intelligent Systems Design and Applications (ISDA'05), 2005.

151. Tokinaga, S., Lu, J., Ikeda, Y., Neural Network Rule Extraction By Using The Genetic Programming and Its Applications to Explanatory Classifications, IECE Trans. Fundamentals, E88-A(10), 2627- 2635, 2005.

152. Markowska-Kaczmar, U., Trelak, W., Fuzzy Logic and Evolutionary Algorithm – Two Techniques in Rule Extraction From Neural Networks, Neurocomputing, 63, 359-379, 2005.

153. Kahramanlı, H., Allahverdi, N., Rule Extraction From Trained Adaptive Neural Networks Using Artificial Immune Systems, Expert Systems with Applications, 36, 1513-1522, 2009.

154. Zhenya, H., Chengjian, W., Luxi, Y., Xiqi, G., Susu, Y., Extracting Rules From Fuzzy Neural Network By Particle Swarm Optimisation, 0-7803-4869-9/98 $10.00©1998 IEEE.

155. Ma, M., Zhou, C., Zhang, L., Dou, Q., Automatic Fuzzy Rule Extraction Based on Particle Swarm Optimization, Proceedings of the Third International Conference on Machine Learning and Cybernetics, Shanghai, 26-29 August 2004.

156. Chen, C., Chen, Y., He, J., Neural Network Ensemble Based Ant Colony Classification Rule Mining, Proceedings of the First International Conference on Innovative Computing, Information and Control (ICICIC'06), 2006.

157. Sivegaminathan, R. K., Ramakrishnan, S., A Hybrid Approach for Feature Subset Selection Using Neural Networks and Ant Colony Optimization, Expert Systems with Applications, 33, 49-60, 2007.

158. Bonabeau, E., Dorigo, M., Theraulaz, G., From Natural to Artificial Swarm Intelligence, Oxford University Press, 1999.

159. Kennedy, J., Eberhart, R. C., Shi, Y., Swarm Intelligence, Academic Press, San Diego, CA, USA, 2001.

160. Grasse, P. P., Termitologia, Tome II., Fondation des Societes, Construction, Paris:Mason, 1984.

161. Dorigo, M., Maniezzo, V., Colorni, A., Positive Feedback As A Search Strategy, Technical Report, 91-016, Dipartmentio di Eletronica, Politecnico di Mileno, Italy, 1991.

162. Dorigo, M., Maniezzo, V., Colorn,i A., The Ant System: Optimization By A Colony of Cooperating Agents, IEEE Transactions on Systems, Man and Cybernetics –Part B, 26 (1), 1-13, 1996.

163. Gambardella, L. M., Taillard, E. D., Dorigo, M., Ant Colonies for the QAP, Technical Report, IDSIA, Lugano, Switzerland, 1997.

164. Deneubourg, J-L., Aron, S., Goss, S., Pasteels, J-M., The Self-Organizing Exploratory Pattern of the Argentine Ant, Journal of Insect Behavior, 3, 159-168, 1990.

165. Engelbrecht, A. P., Fundamentals of Computational Swarm Intelligence, John Wiley & Sons Ltd., England, 2005.

166. Goss, S., Aron, S., Deneubarg, J. L., Pasteels, J. M., Self Organized Shortcuts in The Argentina Ant, Naturwissenchaften, 76, 579-581, 1989.

167. Dorigo, M., Di Caro, G., The Ant Colony Optimization Meta-Heuristic, In D. Corne, M. Dorigo, and F. Glover (Eds.): New Ideas in Optimization, 11-32, McGraw-Hill, 1999.

168. Dorigo, M., Di Caro, G., New Ideas in Optimization, Corne D., Dorigo M., Glover F., McGraw-Hill, Maidenhead, 1-8, 1999.

169. Dorigo, M., Gambardella, L. M., Ant Colony System: A Cooperative Learning Approach to The Travelling Salesman Problem, IEEE Transactions on Evolutionary Computation, 1 (1), 53-66, 1997.

170. Ekin, E., Formal Methods and Programming Tools for Modelling Ant Colonies, Doktora Tezi, Dokuz Eylül Üniversitesi, İzmir, 2006.

171. Stützle, T., Hoos, H. H., The Max-Min Ant System and Local Search for the Travelling Salesman Problem, In T. Baeck, Z. Michalewicz, X. Yao (Eds.): Proceedings of the IEEE International Conference on Evolutionary Computation (ICE'97), 309-314, 1997.

172. Gambardella, L. M., Dorigo, M., Ant-Q: A Reinforcement Learning Approach to the TSP, In Proceedings of ML-95, Twelfth International Conference on Machine Learning, 252-260, 1995.

173. Dorigo, M., Stützle, T., Ant Colony Optimization, MIT Press, Boston, 2004.

174. Wodrich, M., Ant Colony Optimization, Under Graduate Theisis, Department of Electrical and Electronic Engineering, University of Cape Town, South Africa, 1996.

175. Bilchev, G., Parmee, I., Constrained Optimisation with Ant Colony Search Model, Proceedings of ACED96, Plymouth Engineering Design Center, University of Plymouth PL4, 8AA, UK, 1996.

176. Monmarche, N., Venturini, G., Slimane, M., There How Pachycondyla Apicalis Ants Suggest is New Search Algorithm, Future Generation Systems Computer, 16 (8), 937-946, 2000.

177. Karaboğa, N., Kalinli, A., Karaboğa, D., Designing Digital IIR Filters Using Ant Colony Optimisation Algorithm, Engineering Applications of Artificial Intelligence, 17, 301-309, 2004.

178. http://mlearn.ics.edu//MLRepository.html

179. Johansson, U., Löfström, T., König, R., Why Not Use An Oracle When You Got One?, Neural Information Processing –Letters and Reviews, 10 (8-9), 2006.

180. Ross, P. J., Taguchi Techniques For Quality Engineering, McGraw-Hill, New York, 1996.

181. Antony, J., Perry, D., Wang, C., Kumar, M., An Application of Taguchi Method of Experimental Design For New Product Design and Development Process, Assembly Automation 26 (1), 18-24, 2006.

182. Roy, R. K., A Primer on the Taguchi Method, United States of America, New York, 1990.

183. Frank, E., Witten. I. H., Generating Accurate Rule Sets Without Global Optimization. In: Shavlik J. (Ed.), Machine Learning: Proceedings of the 15th International Conference., Morgan Kaufmann Publishers ,144-151, 1998.

184. Kohavi, R. The Power of Decision Tables, In: Lavrac, N., Wrobel, S. (Eds.) Machine Learinng: Proceedings of the 8th European Conference on Machine Learning (ECML95), Lecture Notes in Artificial Intelligence, Springer-Verlag, 914, 174-189, 1995.

185. John, G. H., Langley, P., Estimating Continuous Distributions in Bayesian Classifiers, In: Proceedings of the 11th Conference on Uncertainty in Artificial Intelligence, Morgan Kaufmann, San Mateo, 338-345, 1995.

186. Öztuna, D., Elhan, A. H., Gruplararası ve Grupiçi Karşılaştırma Yöntemleri, Toroks Derneği 8. Yıllık Kongresi, Kemer/Antalya, 27 Nisan-1 Mayıs 2005.

187. Özbakır, L., Baykasoğlu, A., Kulluk, S., Yapıcı, H., TACO-miner: An Ant Colony Based Algorithm for Rule Extraction from Trained Neural Networks, Expert Systems with Applications, submitted.

188. Özbakır, L., Baykasoğlu, A., Kulluk, S., Yapıcı, H., Rule Extraction from Neural Networks for Discovering Causes of Quality Defects in Fabric Production, Engineering Applications of Artificial Intelligence, submitted.

189. Özbakır, L., Baykasoğlu, A., Kulluk, S., Tapkan, P. Z.., Data Mining Approach to Discover Causes of Quality Defects in Fabric Production, HDM 2008, Kayseri/TURKEY, 19-23 June 2008.

190. Kulluk, S., Özbakır, L., Baykasoğlu, A., Gürbüz, F., Karınca Koloni Optimizasyonu ile Yapay Sinir Ağlarından Sınıflandırma Kuralları Çıkarımı,

YA/EM72007 Yöneylem Araştırması/Endüstri Mühendisliği XXVII. Ulusal Kongresi, İzmir, 2-4 Temmuz 2007.

ÖZGEÇMİŞ

Sinem KULLUK 1977 yılında Kayseri'de doğdu. İlk, orta ve lise öğrenimini Kayseri'de tamamladı. 1996 yılında Erciyes Üniversitesi, Mühendislik Fakültesi, Endüstri Mühendisliği bölümünü kazandı ve 2000 yılında mezun oldu. 2003 yılında Gazi Üniversitesi, Fen Bilimleri Enstitüsü, Endüstri Mühendisliği Anabilim Dalında yüksek lisans eğitimini tamamladı. Aynı yıl Erciyes Üniversitesi, Fen Bilimleri Enstitüsü, Makine Mühendisliği Anabilim Dalında doktora eğitime başladı. 2001 yılı Aralık ayında Erciyes Üniversitesi, Mühendislik Fakültesi, Endüstri Mühendisliği Bölümü'nde araştırma görevlisi olarak göreve başladı ve halen aynı bölümde görevine devam etmektedir.

İletişim Bilgileri:
Erciyes Üniversitesi, Mühendislik Fakültesi,
Endüstri Mühendisliği Bölümü,
38039 KAYSERİ
Tel: (0 352) 4374901 / 32452
e-mail: skulluk@erciyes.edu.tr.

EKLER

EK-1 7. BÖLÜM TEST PROBLEMLERİNDE ELDE EDİLEN KURALLAR

Ek-1.1. CRX veri kümesinde çıkarılan örnek veri kümesi

Kural No	Kalite	Uygunluk	Elde edilen örnek kural kümesi
1	0,9999	0,6716	Eğer (A4=u veya y) ve (A6=w veya q veya r veya cc veya k veya c veya x veya i veya e veya aa veya j) ve (A7=v veya h veya bb veya n) ve (A9=t) ise sınıf "+"
2	0,9999	0,5728	Eğer (A5=g veya p) ve (A6=w veya q veya m veya k veya c veya d veya x veya j) ve (A7=v veya h veya bb veya j veya z veya o veya dd veya n) ve (A9=t) ise sınıf "+"
3	0,9999	0,5038	Eğer (A5=g) ve (A6=w veya q veya r veya cc veya k veya c veya x veya i veya e veya aa veya j) ve (A7=v veya h veya bb veya j veya z veya o veya d) ve (A8>1.02) ve (A9=t) ise sınıf "+"
4	0,9999	0,5009	Eğer (A4=u veya y) ve (A6= w veya k veya c veya d veya x veya e veya aa veya ff veya j) ve (A7=v veya bb veya ff veya j veya z veya d veya n) ve (A9=f) ise sınıf "-".
5	0,9999	0,4972	Eğer (A6=w veya q veya r veya cc veya k veya c veya x veya i veya e veya aa veya j) ve (A7=v veya h veya bb veya ff veya o veya d) ve (A10=t) ise sınıf "+".
6	0,9999	0,486	Eğer (A4= u veya y) ve (A6= q veya m veya r veya cc veya c veya d veya x veya i veya aa veya ff veya j) ve (A7= v veya bb veya ff veya j veya z veya n) ve (A11≤2.5) ve (A13=g veya s) ise sınıf "-".
7	0,9999	0,471	Eğer (A6=w veya q veya r veya cc veya k veya c veya x veya i veyae veya aa veya j) ve (A7=v veya h veya bb veya j veay z veya o veya d) ve (A8>1.02) ise sınıf "+".
8	0,9989	0,456	Eğer (A6=w veya q veya c veya d veya x veya i veya e veya aa veya ff veya j) ve (A7=v veya h veya bb veya ff veya j veya z veya n) ve (A11≤2.5) ve (A13=g veya s) ise sınıf "-".

9	0,9999	0,4464	Eğer (A5=g) ve (A6=w veya q veya r veya cc veya k veya c veya x veya i veya e veya aa veya j) ve (A7=v veya h veya bb veya j veya z veya o veya d) ve (A10=t) ve (A13=g) ise sınıf "+".
10	0,9999	0,4264	Eğer (A6= q veya c veya x veya i veya e veya aa veya ff veya j) ve (A7=v veya h veya bb veya ff veya j veya z veya o veya n) ve (A11≤2.5) ve (A13=g veya s) ise sınıf "-".
11	0,9999	0,4253	Eğer (A5= g) ve (A6=w veya q veya r veya cc veya k veya c veya x veya i veya aa veya ff veya j) ve (A7= v veya h veya bb veya ff veya o veya d veya n) ve (A10=t) ve (A13=g) ise sınıf"+".
12	0,9999	0,4247	Eğer (A6=w veya k veya c veya x veya i veya e veya aa veya ff veya j) ve (A7=v vye h veya bb veya ff veya j veya z veya n) ve (A11≤2.5) ve (A14>105) ise sınıf "-".
13	0,9981	0,4125	Eğer (A4=u veya y) ve (A6=q veya m veya r veya cc veya c veya d veya x veya i veya aa veya ff veya j) ve (A7= v veya bb veyaff veya j veya n) ve (A8≤1.02) ise sınıf "-".
14	0,9999	0,4044	Eğer (A4=u veya y) ve (A6=q veya m veya r veya cc veya c veya d veya x veya i veya aa veya ff veya j) ve (A7=v veya h veya j veya z veya n) ve (A11≤2.5) ve (A13=g veya s) ve (A15≤492) ise sınıf "-".
15	0,9999	0,3899	Eğer (A6=w veya q veya r veya k veya c veya d veya e veya aa veya ff) ve (A7=v veya bb veya ff veya j veya n) ve (A8≤1.02) ise sınıf "-".
16	0,9958	0,3827	Eğer (A6=w veya q veya m veya k veya c veya d veya e veya aa veya ff) ve (A7=v veya h veya bb veya ff veya o veya d) ve (A8>1.02) ise sınıf "+".
17	0,9999	0,3685	Eğer (A4= u veya y) ve (A6=w veya m veya k veya c veya e veya aa veya ff) ve (A7=v vye bb veya ff veya z veya n) ve (A8≤1.02) ise sınıf "-".
18	0,9999	0,3626	Eğer (A6=w veya q veya k veya c veya d veya x veya i veya e veya ff veya j) ve (A7=v veya h veya z veya d) ve (A9=f) ve (A10=f) ve (A11≤2.5) ve (A13=g veya s) ise sınıf "-".
19	0,9999	0,3624	Eğer (A4=u veya y) ve (A5=g) ve (A6=w veya q veya r veya cc veya k veya c veya x veya i veya e veya aa veya j) ve (A7=v veya h veya bb veya j veya z veya o veya d) ise sınıf "+".
20	0,9487	0,3546	Eğer (A2≤38.96) ve A4=u veya y) ve (A6=w veya c veya d veya x veya i veya aa veya ff veya j) ve (A7=v veya bb veya ff veya j veya z veya n) ve (A13=g veya s) ve (A15≤492) ise sınıf "-".

21	0,9999	0,3543	Eğer (A4= u veya y) ve (A6=w veya cc veya c veya x veya i veya e veya aa) ve (A7= v veya h veya bb veya ff veya d veya n) ve (A9=f) ve (A13=g) ve (a15≤492) ise sınıf "-".
22	0,9745	0,3476	Eğer (A2≤38.96) ve (A4= u veya y) ve (A6=q veya m veya r veya cc veya c veya d veya x veya i veya aa veya ff veya j) ve (A7=v veya bb veya ff veya j veya z veya n) ise sınıf "-".
23	0,9999	0,3346	Eğer (A2≤38.96) ve (A4= u veya y) ve (A6=q veya m veya r veya cc veya k veya x veya i veya e veya aa veya ff veya j) ve (A7=ve veya h veya ff veya j veya z veya d) ve (A15≤492) ise sınıf "-".
24	0,9999	0,3199	Eğer (A6=w veya q veya m veya cc veya k veya c veya x veya i veya e veya aa veya ff) ve (A7=h veya bb veya ff veya n) ve (A9=t) ise sınıf "+".
25	0,9999	0,3092	Eğer (A6=w veya q veya m veya r veya c veya x veya aa veya j) ve (A7=v veya h veya j veya z veya n) ise sınıf "+".
26	0,9999	0,3064	Eğer (A6=w veya q veya m veya x veya i veya e veya aa veya j) ve (A7=v veya h veya bb veya j veya z veya o veya d) ise sınıf "+".
27	0,9999	0,2269	Eğer (A4=u veya y) ve (A6=w veya k veya x veya i veya aa veya ff veya j) ve (A7=v veya h veya z) ve (A13=g veya s) ise sınıf "-".
28	0,9999	0,2032	Eğer (A4= u veya y) ve (A6=q veya m veya r veya cc veya c veya d veya x veya i) ve (A7=v veya h veya z veya n) ve (A11≤0.5 veya A11≥2.5) ise sınıf "-".
29	0,9999	0,1335	Eğer (A6=r veya cc veya c veya aa veya ff) ve (A7=h veya bb veya ff veya o veya d) ise sınıf "+".

Eğitim Doğruluğu:	98,71	Güvenilirlik:	100
Test Doğruluğu:	98,55	Destek:	98,55
ÇKA En İyi MSE değeri:	0,03	Genellik:	98,55

Ek-1.2. Heart-C veri kümesinde çıkarılan örnek veri kümesi

Kural No	Kalite	Uygunluk	Elde edilen örnek kural kümesi
1	0,9999	0,5494	Eğer (cp=4) ve (thal=7) ise sınıf "3"
2	0,9997	0,537	Eğer (thalach≥114.67) ve (slope=1 veya 2) ve (ca=0) ise sınıf "0"
3	0,9999	0,4551	Eğer (cp=2 veya 3) ve (chol<418) ve (rectecg=0 veya 2) ve (slope=1 veya 2) ve (ca=0 veya 2) ise sınıf "0".
4	0,9999	0,4513	Eğer (cp=2 veya 3) ve (chol<418) ve (exang=no) ve (oldpeak<4.133) ve (ca=0 veya 3) ise sınıf "0".
5	0,9724	0,4499	Eğer (age<61) ve (chol≤272) ve (restecg=0 veya 2) ve (exang=no) ve (oldpeak<4.133) ve (ca=0 veya 3) ise sınıf "0".
6	0,9999	0,4425	Eğer (restecg=2) ve (thal=7) ise sınıf "4".
7	0,9992	0,4357	Eğer (cp=1 veya 2 veya 3) ve (chol≤272) ve (restecg=0 veya 2) ve (exang=no) ise sınıf "0".
8	0,9868	0,4314	Eğer (age≥45) ve (cp=4) ve (thalach<158.33) ise sınıf "2".
9	0,9999	0,4302	Eğer (cp= 1 veya 2 veya 3) ve (chol<418) ve (restecg=0 veya 2) ve (slope=1) ve (ca=0 veya 1 veya 3) ise sınıf "0".
10	0,9994	0,4289	Eğer (age<61) ve (cp=1 veya 2 veya 4) ve (thalach<158.33) ise sınıf "2".
11	0,9955	0,4266	Eğer (age≥45) ve (cp=2 veya 4) ve (thalach<158.33) ise sınıf "2".
12	0,9999	0,418	Eğer (exang=0 veya 2) ise sınıf "0".
13	0,9999	0,4161	Eğer (cp=2 veya 3) ve (chol<272) ve (ca=0 veya 2 veya 3) ise sınıf "0".
14	0,9948	0,404	Eğer (age<61) ve (cp=1 veya 4) ve (trestbps<164.67) ve (thal=6 veya 7) ise sınıf "2".
15	0,994	0,4006	Eğer (age≥45) ve (thalach<158.33) ve (thal=6 veya 7) ise sınıf "2".
16	0,9975	0,3594	Eğer (cp=1 veya 3 veya 4) ve (fbs=false) ve (restecg=2) ve (2.067≤oldpeak<4.133) ise sınıf "4".
17	0,9971	0,3229	Eğer (sex=female) ve (restecg=0 veya 2) ve (exang=no) ve (ca=0 veya 2) ise sınıf "0".
18	0,9999	0,3076	Eğer (sex=female) ve (cp=1 veya 2 veya 3) ve (exang=no) ve (oldpeak<4.133) ise sınıf "0".
19	0,9239	0,3009	Eğer (chol<418) ve (restecg=1 veya 2) (thalach<158.33) ve (oldpeak<4.133) ve (ca=0 veya 2 veya 3) ve (thal=3 veya

			7) ise sınıf “3”.
20	0,9999	0,2941	Eğer (age<61) ve (slope=1 veya 2) ve (ca=2 veya 3) ise sınıf “3”.
21	0,9999	0,2883	Eğer (45≤age<61) ve (chol<272) ve (oldpeak<4.133) ve (ca=0 veya 1 veya 3) ve (thal=3 veya 7) ise sınıf “1”.
22	0,996	0,2838	Eğer (age<45 ve age≥61) ve (cp=1 veya 3 veya 4) ve (tretsbps<164.67) ve (restecg=2) ve (ca=1 veya 2 veya 3) ise sınıf “4”.
23	0,996	0,2822	Eğer (45≤age<61) ve (cp=1 veya 2 veya 4) ve (restecg=1 veya 2) ve (thal=3 veya 7) ise sınıf “1”.
24	0,9744	0,2796	Eğer (45≤age<61) ve (cp=4) ve (oldpeak<4.133) ve (thal=3 veya 7) ise sınıf “1”.
25	0,9928	0,2783	Eğer (cp=1 veya 3 veya 4) ve (thalach<158.33) ve (oldpeak<4.133) ve (ca=0 veya 3) ve (thal=3 veya 7) ise sınıf “1”.
26	0,9972	0,2575	Eğer (age≥45) ve (chol<272) ve (oldpeak<4.133) ve (ca=1 veya 2 veya 3) ve (thal=3 veya 7) ise sınıf “1”.
27	0,9999	0,257	Eğer (trestbps<164.67) ve (restecg=2) ve (ca=0 veya 2) ise sınıf “0”.
28	0,9679	0,2477	Eğer (cp=1 veya 3 veya 4) ve (restecg=1 veya 2) ve (thalach<158.33) ve (oldpeak<4.133) ve (thal=3 veya 7) ise sınıf “1”.
29	0,9511	0,2397	Eğer (cp=1 veya 3 veya 4) ve (trestbps<164.67) ve (restecg=1 veya 2) ve (oldpeak<4.133) ve (ca=1 veya 2 veya 3) ise sınıf “1”.
30	0,9999	0,2195	Eğer (age<61) ve (sex=male) ve (cp=1 veya 2 veya 3) ve (exang=no) ve (thal=3 veya 6) ise sınıf “0”.
31	0,9518	0,205	Eğer (sex=male) ve (thalach≥158.33) ve (thal=3 veya 7) ise sınıf “1”.
32	0,9999	0,1758	Eğer (sex=male) ve (cp=3 veya 4) ve (chol<272 veya chol≥418) ve (restecg=0) ve (slope=1 veya 2) ve (ca=1 veya 2 veya 3) ve (thal=7) ise sınıf “3”.
33	0,9802	0,1742	Eğer (age≥61) ve (ca=2 veya 3) ve (thal =3 veya 7) ise sınıf “3”.
34	0,9879	0,1533	Eğer (age<45 veya age≥61) ve (cp=1 veya 3 veya 4) ve (trestbps<129.33) ve (thalach≥158.33) ve (exang=yes) ve (ca=0 veya 3) ise sınıf “4”.

35	0,9999	0,1319	Eğer (sex=female) ve (trestbps<129.22 veya trestbps≥164.67) ve (chol<418) ve (slope=1) ise sınıf "0".
36	0,9728	0,124	Eğer (sex=male) ve (272≤chol<418) ve (oldpeak<2.067 veya oldpeak≥4.133) ve (slope=2 veya 3) ve (ca=1 veya 3) ve (thal=7) ise sınıf "3".
37	0,9907	0,1234	Eğer (cp=4) ve (fbs=false) ve (restecg=1 veya 2) ve (thalach<114.67) ve (exang=yes) ve (thal=3 veya 7) ise sınıf "3".
38	0,9884	0,0498	Eğer (trestbps<129.33 veya trestbps≥164.67) ve (chol<272 veya chol≥418) ve (exang=yes) ve (ca=0 veya 1 veya 2) ise sınıf "0".

Eğitim Doğruluğu:	99,26	Güvenilirlik:	100
		Destek:	100
Test Doğruluğu:	100	Genellik:	100
ÇKA En İyi MSE değeri:	0,053		

Ek-1.3. LBC veri kümesinde çıkarılan örnek veri kümesi

Kural No	Kalite	Uygunluk	Elde edilen örnek kural kümesi
1	0,9999	0,3885	Eğer (age={30-39} veya {40-49} veya {50-59} veya {60-69}) ve (tumorsize={0-4} veya {5-9} veya{10-14} veya {15-19} veya {20-24} veya {25-29} veya {35-39} veya {40-44}) ve (invnodes={0-2} veya {3-5} veya {6-8} veya {9-11} veya {15-17}) ve (degmalig=1 veya 2) ve (irradiat=no) ise sınıf "no-recurrence-events".
2	0,9972	0,3635	Eğer (age={30-39} veya {40-49} veya {50-59} veya {60-69} veya {70-79}) ve (invnodes={0-2} veya {12-14}) ise sınıf "no-recurrence-events".
3	0,9937	0,3575	Eğer (age={30-39} veya {40-49} veya {50-59} veya {60-69} veya {70-79}) ve (invnodes={0-2} veya {15-17}) ise sınıf "no-recurrence-events".
4	0,9999	0,3533	Eğer (age={30-39} veya {40-49} veya {50-59} veya {60-69}) ve (tumorsize={0-4} veya {5-9} veya{10-14} veya {15-19} veya {20-24} veya {25-29} veya {35-39} veya {40-44}) ve (invnodes={0-2} veya {3-5} veya {6-8} veya {9-11}) ve (nodecaps=no) ve (breastquad=left_low veya left_up) ise sınıf "no-recurrence-events".
5	0,9945	0,3448	Eğer (age={30-39} veya {40-49} veya {50-59} veya {60-69}) ve (tumorsize={0-4} veya {10-14} veya {15-19} veya {20-24} veya {35-39} veya {40-44}) ve (invnodes={0-2} veya {6-8} veya {9-11}) ise sınıf "no-recurrence-events".
6	0,9999	0,3199	Eğer (age={50-59} veya {60-69}) ve (breastquad=left_low veya left_up) ise sınıf "no-recurrence-events".
7	0,9999	0,3085	Eğer (age={30-39} veya {40-49} veya {50-59} veya {60-69} veya {70-79}) ve (tumorsize={10-14} veya {20-24} veya {30-34} veya {40-44}) ve (invnodes={0-2} veya veya {6-8} veya {12-14} veya {24-26}) ise sınıf "no-recurrence-events".
8	0,9999	0,3031	Eğer (age={20-29} veya {30-39} veya {40-49} veya {50-59} veya {60-69}) ve (menopause=premeno veya ge40) ve (tumorsize={0-4} veya {10-14} veya {15-19} veya {20-24} veya {35-39} veya {40-44} veya {45-49}) ve (degmalig= 1 veya 2) ve (breastquad=left_low veya right_up veya left_up veya central) ve (irradiat=no) ise sınıf "no-recurrence-events".
9	0,9999	0,2957	Eğer (age={40-49} veya {50-59} veya {60-69} veya {70-79}) ve (tumorsize={0-4} veya {5-9} veya {15-19} veya {25-29} veya {30-34} veya {35-39} veya {40-44} veya {45-49} veya {50-54}) ve (breastquad=left_low veya

			right_up veya right_low veya central) ise sınıf "recurrence-events".
10	0,9999	0,2921	Eğer (age={30-39} veya {40-49} veya {50-59}) ve (tumorsize={0-4} veya {5-9} veya{10-14} veya {15-19} veya {20-24} veya {25-29} veya {35-39} veya {40-44}) ve (invnodes={0-2} veya {3-5} veya {6-8} veya {9-11}) ve (degmalig=1 veya 2) ve (breastquad=left_low veya right_up veya left_up) ise sınıf "no-recurrence-events".
11	0,9999	0,2834	Eğer (age={40-49} veya {50-59} veya {60-69} veya {70-79}) ve (tumorsize={0-4} veya {5-9} veya {15-19} veya {25-29} veya {30-34} veya {35-39} veya {40-44} veya {45-49} veya {50-54}) ve (invnodes={0-2} veya {3-5} veya {9-11} veya {12-14} veya {15-17} veya {24-26}) ve (degmalig=3) ise sınıf "recurrence-events".
12	0,9773	0,2826	Eğer (age={40-49} veya {50-59} veya {60-69} veya {70-79}) ve (tumorsize={0-4} veya {15-19} veya {25-29} veya {30-34} veya {50-54}) ve (invnodes={0-2} veya {3-5} veya {9-11} veya {12-14} veya {15-17}) ise sınıf "recurrence-events".
13	0,9999	0,2818	Eğer (age={40-49} veya {50-59} veya {60-69} veya {70-79}) ve (tumorsize={0-4} veya {5-9} veya {20-24} veya {25-29} veya {30-34} veya {35-39} veya {40-44} veya {45-49} veya {50-54}) ve (invnodes={0-2} veya {3-5} veya {9-11} veya {12-14} veya {15-17} veya {24-26}) ve (breastquad=letf_low veya right_up veya right_low veya central) ise sınıf "recurrence-events".
14	0,992	0,2801	Eğer (menopause=premeno veya ge40) ve (tumorsize={15-19} veya {30-34} veya {40-44} veya {45-49}) ve (invnodes={0-2} veya {3-5} veya {6-8} veya {9-11}) ise sınıf "recurrence-events".
15	0,9863	0,2776	Eğer (age={20-29} veya {30-39} veya {40-49} veya {50-59} veya {60-69}) ve (tumorsize={0-4} veya {5-9} veya{10-14} veya {15-19} veya {20-24} veya {35-39} veya {40-44}) ve (invnodes={0-2} veya {3-5} veya {6-8} veya {9-11} veya {12-14} veya {24-26}) ve (degmalig=2 veya 3) ise sınıf "no-recurrence-events".
16	0,9999	0,2704	Eğer (tumorsize={10-14} veya {20-24} veya {25-29} veya {30-34} veya {45-49} veya {50-54}) ve (invnodes={0-2} veya {3-5} veya {9-11} veya {12-14} veya {15-17}) ve (breastquad=right_up veya left_up) ise sınıf "recurrence-events".
17	0,9953	0,2684	Eğer (menopause=premeno veya lt40) ve (invnodes={0-2} veya {3-5} veya {9-11} veya {12-14} veya {15-17}) ise

			sınıf “recurrence-events”.
18	0,9999	0,2388	Eğer (age={30-39} veya {40-49} veya {60-69}) ve(menopause=premeno veya ge40) ve (tumorsize={0-4} veya {15-19} veya {25-29} veya {30-34} veya {40-44} veya {45-49}) ve (invnodes={0-2} veya {6-8} veya {9-11}) ise sınıf “recurrence-events”.
19	0,9526	0,2358	Eğer (age={20-29} veya {30-39} veya {40-49} veya {50-59} veya {60-69}) ve (tumorsize={0-4} veya {10-14} veya {15-19} veya {20-24} veya {25-29} veya {35-39} veya {40-44} veya {45-49}) ve (breastquad=left_low veya central) ve (irradiat=no) ise sınıf “no-recurrence-events”.
20	0,9939	0,2325	Eğer (age={20-29} veya {30-39} veya {40-49} veya {70-79}) ve(menopause=premeno) ve (tumorsize={0-4} veya {5-9} veya{10-14} veya {15-19} veya {20-24} veya {25-29} veya {30-34} veya {35-39} veya {40-44}) ve (invnodes={0-2} veya {3-5} veya {6-8} veya {9-11} veya {12-14} veya {24-26}) ise sınıf “no-recurrence-events”.
21	0,9999	0,2321	Eğer (age={40-49} veya {50-59} veya {60-69} veya {70-79}) ve (tumorsize={10-14} veya {20-24} veya {25-29} veya {30-34} veya {35-39} veya {40-44} veya {45-49} veya {50-54}) ve (invnodes={0-2} veya {3-5} veya {6-8} veya {9-11} veya {15-17}) ve (breastquad=right_up veya left_up veya central) ise sınıf “recurrence-events”.

Eğitim Doğruluğu:	100	Güvenilirlik:	100
Test Doğruluğu:	100	Destek:	100
ÇKA En İyi MSE değeri:	0,106	Genellik:	100

Ek-1.4. Monks-2 veri kümesinde çıkarılan örnek veri kümesi

Kural No	Kalite	Uygunluk	Elde edilen örnek kural kümesi
1	0,9946	0,3071	Eğer (A3=1) ve (A6=2) ise sınıf "1".
2	0,9999	0,3018	Eğer (A2=1 veya 3) ve (A6=1) ise sınıf "0".
3	0,9999	0,3	Eğer (A6=1) ise sınıf "0".
4	0,9999	0,299	Eğer (A2=1 veya 3) ve (A4=1 veya 2) ise sınıf "0".
5	0,9999	0,2853	Eğer (A4=1 veya 2) ise sınıf "0".
6	0,9913	0,2844	Eğer (A4=2 veya 3) ise sınıf "1".
7	0,9999	0,2813	Eğer (A4=1 veya 3) ve (A5=1 veya 2 veya 4) ise sınıf "0".
8	0,9999	0,276	Eğer (A1=2 veya 3) ve (A3=2) ve (A5=2 veya 3 veya 4) ise sınıf "0".
9	0,9018	0,2738	Eğer (A5= 2 veya 3) ve (A6=2) ise sınıf "1".
10	0,9999	0,2545	Eğer (A1=1) ise sınıf "0".
11	0,9078	0,2455	Eğer (A1=2 veya 3) ve (A2=1 veya 3) ve (A5=2 veya 3 veya 4) ise sınıf "1".
12	0,9999	0,2455	Eğer (A4=1 veya 3) ve (A5=1 veya 2 veya 3) ise sınıf "0".
13	0,9859	0,1473	Eğer (A4=1) ve (A6=2) ise sınıf "1".
14	0,9896	0,1265	Eğer (A3=2) ve (A4=1) ise sınıf "1".

Eğitim Doğruluğu:	100	Güvenilirlik:	100
Test Doğruluğu:	100	Destek:	100
ÇKA En İyi MSE değeri:	0,01	Genellik:	100

Ek-1.5. Nursery veri kümesinde çıkarılan örnek veri kümesi

Kural No	Kalite	Uygunluk	Elde edilen örnek kural kümesi
1	0,9761	1	Eğer (parents=usual) ve (has_nurs=proper) ve (form=complete) ve (children= 1) ve (housing=convenient) ve (finance=convenient) ve (health=recommended) ise sınıf "2".
2	0,9999	0,9336	Eğer (parents=usual veya pretentious) ve (has_nurs=proper veya less_proper veya improper) ve (social=nonprob veya slightly_prob) ve (health=recommended) ise sınıf "3".
3	0,9999	0,6686	Eğer (parents= pretentious veya great_pret) ve (health= not_recom) ise sınıf "1".
4	0,9999	0,5551	Eğer (has_nurs= improper veya critical veya very_crit) ve (form=complete veya completed veya foster) ve (health=recommended veya priority) ise sınıf "5".
5	0,9999	0,5248	Eğer (has_nurs=proper veya less_proper veya improper veya critical) ve (housing=convenient veya less_conv) ve (health=recommended veya priority) ise sınıf "4".
6	0,9999	0,5099	Eğer (parents=usual veya pretentious) ve (has_nurs=proper veya less_proper veya improper veya critical) ve (form=complete veya completed veya incomplete) ve (health=recommended veya priority) ise sınıf "4".
7	0,9991	0,5021	Eğer (health= priority veya not_recom) ise sınıf "1".
8	0,9999	0,5013	Eğer (has_nurs=proper veya less_proper veya improper veya critical) ve (housing=convenient veya critical) ve (health=recommended veya priority) ise sınıf "4".
9	0,9767	0,4842	Eğer (housing=convenient veya less_conv) ve (health=recommended veya priority) ise sınıf "4".
10	0,9999	0,4837	Eğer (parents= pretentious veya great_pret) ve (has_nurs= improper veya critical veya very_crit) ve (housing=less_conv veya critical) ve (health=recommended veya priority) ise sınıf "5".
11	0,9999	0,4696	Eğer (parents= pretentious veya great_pret) ve (has_nurs=proper veya improper veya critical veya very_crit) ve (social= slightly_prob veya problematic) ve (health=recommended veya priority) ise sınıf "5".
12	0,9999	0,4402	Eğer (parents=usual veya pretentious) ve (housing=convenient veya critical) ve (health=recommended veya priority) ise sınıf "4".
13	0,9999	0,4251	Eğer (parents= pretentious veya great_pret) ve (has_nurs=proper veya improper veya critical veya

			very_crit) ise sınıf “5”.
14	0,9254	0,3782	Eğer (housing=convenient veya critical) ve (social=nonprob veya slightly_prob) ve (health=recommended veya priority) ise sınıf “4”.
15	0,9999	0,3729	Eğer (parents= pretentious veya great_pret) ve (has_nurs= less_proper veya improper veya critical veya very_crit) ve (form=complete veya incomplete veya foster) ve (children= 1 veya 2 veya more) ve (health=recommended veya priority) ise sınıf “5”.
16	0,9999	0,3546	Eğer (form= completed veya incomplete veya foster) ve (children= 2 veya 3 veya more) ve (social= slightly_prob veya problematic) ve (health=recommended veya priority) ise sınıf “5”.
17	0,9998	0,3487	Eğer (has_nurs=proper veya improper veya very_crit) ve (form=complete veya completed veya incomplete) ve (health=recommended veya priority) ise sınıf “5”.
18	0,9999	0,3336	Eğer (has_nurs=proper veya improper veya critical veya very_crit) ve (form= incomplete veya foster) ve (children= 1 veya 2 veya more) ve (health=recommended veya priority) ise sınıf “5”.
19	0,9999	0,2818	Eğer (form= completed veya incomplete veya foster) ve (housing= less_conv veya critical) ve (social=slightly_prob veya problematic) ise sınıf “5”.
20	0,9146	0,2665	Eğer (form= completed veya foster) ve (housing= less_conv veya critical) ise sınıf “5”.
21	0,9999	0,2643	Eğer (parents= pretentious veya great_pret) ve (has_nurs=proper veya less_proper veya improper veya very_crit) ve (social=nonprob veya problematic) ise sınıf “5”.
22	0,9999	0,2475	Eğer (form= incomplete veya foster) ve (children=3 veya more) ve (health=recommended veya priority) ise sınıf “5”.
23	0,9999	0,2389	Eğer (parents= pretentious veya great_pret) ve (has_nurs=proper veya less_proper veya very_crit) ve (form=complete veya incomplete veya foster) ve (health=recommended veya priority) ise sınıf “5”.

Eğitim Doğruluğu:	98,71	Güvenilirlik:	100
Test Doğruluğu:	98,55	Destek:	100
ÇKA En İyi MSE değeri:	0,025	Genellik:	100

Ek-1.6. Pima veri kümesinde çıkarılan örnek veri kümesi

Kural No	Kalite	Uygunluk	Elde edilen örnek kural kümesi
1	0,9999	0,5018	Eğer (A1<12.75) ve (A2<149.25) ve (A3≥30.5) ve (A8<36) ise sınıf "0".
2	0,9532	0,4961	Eğer (A1<12.75) ve (A2<149.25) ve (A3≥30.5) ve (A5<211.5) ve (A8<36 veya A8≥51) ise sınıf "0".
3	0,9999	0,4918	Eğer (A2<149.25) ve (A8<36) ise sınıf "0".
4	0,9999	0,4902	Eğer (A2<149.25) ve (A8<36 veya A8≥51) ise sınıf "0".
5	0,9574	0,4745	Eğer (A1<4.25) ve (49.75≤A2<149.25) ve (A3≥30.5) ve Eğer (A5<425 veya A5≥634.5) ve (A6<50.325) ve (A7<1.8345) ise sınıf "0".
6	0,9999	0,4658	Eğer (A1<12.75) ve (A2<149.25) ve (A3≥30.5) ve (A5<211.5 veya A5≥634.5) ve (A7<0.6635) ve (A8<51) ise sınıf "0".
7	0,9587	0,4658	Eğer (A1<8.5) ve (A2<149.25) ve (A3≥30.5) ve (A7<1.8345) ise sınıf "0".
8	0,9999	0,4653	Eğer (A1<4.25 veya A1≥12.75) ve (49.75<A2≤149.25) ve (30.5<A3≤91.5) ve (A7<1.8345) ise sınıf "0".
9	0,958	0,4575	Eğer (A1<8.5) ve (A2<149.25) ve (A4<49.5) ve (A5<211.5) ve (A7<0.6635) ve (A8<51) ise sınıf "0".
10	0,9577	0,4542	Eğer (A1<12.75) ve (A2<149.25) ve (A3≥30.5) ve (A5<211.5 veya A5≥634.5) ve (A6<33.55) ise sınıf "0".
11	0,9537	0,4076	Eğer (A1<8.5) ve (A2≥49.75) ve (A3≥30.5) ve (A5<211.5 veya A5≥634.5) ve (A7<0.6635) ve (A8<36) ise sınıf "0".
12	0,9538	0,4028	Eğer (A1<8.5) ve (A2≥49.75) ve (A3≥30.5) ve (A5<211.5) ve (A8<36 veya A8≥66) ise sınıf "0".
13	0,9955	0,4012	Eğer (A2<149.25) ve (A8<51) ise sınıf "0".
14	0,9554	0,3949	Eğer (A1<8.5 veya A1≥12.75) ve (A2≥49.75) ve (A3≥30.5) ve (A5<211.5 veya A5≥423) ve (A7<0.6635) ve (A8<36 veya 51<A8≤66) ise sınıf "0".
15	0,9999	0,3933	Eğer (A5<211.5) ve (A6<33.55) ve (A7<1.8345) ise sınıf "0".
16	0,9643	0,3791	Eğer (A1<12.75) ve (A3≥30.5) ve (A8<36) ise sınıf "0".
17	0,9999	0,3753	Eğer (A2≥149.25) ise sınıf "1".
18	0,9588	0,3731	Eğer (A1<12.75) ve (A3≥30.5) ve (A5<211.5 veya A5≥634.5) ve (A7<0.6635) ve (A8<51) ise sınıf "0".
19	0,9032	0,353	Eğer (A4<74.25) ve (A5<423) ve (33.55<A6≤50.325) ise

			sınıf "1".
20	0,9999	0,3369	Eğer (A4<74.25) ve (A5<211.5 veya 423<A5≤634.5) ve (A6≥33.55) ve (A7<1.249 veya A7≥1.8345) ve (A8<66) ise sınıf "1".
21	0,934	0,3322	Eğer (A1<8.5 veya A1≥13.75) ve (A3≥30.5) ve (A4<24.75 veya A4≥49.5) ve (A5<423 veya A5≥634.5) ve (A6<50.325) ise sınıf "0".
22	0,9999	0,3229	Eğer (A5<211.5) ve (A6<16.75 veya 33.55<A6≤50.325) ve (A7<1.249 veya A7≥1.8345) ve (A8<66) ise sınıf "1".
23	0,905	0,3158	Eğer (4.25<A1≤12.75) ve (A4<74.25) ve (A5<423) ise sınıf "1".
24	0,9999	0,248	Eğer(A4≥24.75) ve (A5<211.5) ise sınıf "1".
25	0,902	0,2475	Eğer (A2≥99.5) ve (A4<24.75) ve (A7<0.6635 veya A7≥1.8345) ve (A8<66) ise sınıf "1".
26	0,9956	0,2455	Eğer (99.5<A2≤149.25) ve (A5<211.5 veya 423<A5≤634.5) ise sınıf "1".
27	0,9999	0,214	Eğer (99.5<A2≤149.25) ve (A4<49.5) ve (A7<0.6635 veya A7≥1.8345) ve (A8<51) ise sınıf "1".
28	0,9768	0,2124	Eğer (24.75<A4≤74.25) ve (A7<1.249) ve (A8<66) ise sınıf "0".
29	0,9642	0,1979	Eğer (A1<4.25 veya A1≥8.5) ve (A2<99.5 veya A2≥149.25) ve (A3<91.5) ise sınıf "1".

Eğitim Doğruluğu:	98,71	Güvenilirlik:	100
Test Doğruluğu:	98,55	Destek:	100
ÇKA En İyi MSE değeri:	0,18	Genellik:	100

Ek-1.7. Spect-Heart veri kümesinde çıkarılan örnek veri kümesi

Kural No	Kalite	Uygunluk	Elde edilen örnek kural kümesi
1	0,9999	0,5738	Eğer (F13=0) ve (F14=0) ve (F16=0) ve (F17=0) ve (F18=0) ve (F19=0) ve (F20=0) ise sınıf "0".
2	0,9999	0,5625	Eğer (F11=0) ve (F16=0) ve (F17=0) ve (F22=0) ise sınıf "0".
3	0,9589	0,5031	Eğer (F1=0) ve (F7=0) ve (F12=0) ve (F13=0) ve(F16=0) ve (F19=0) ise sınıf "0".
4	0,9999	0,3263	Eğer (F1=1) ise sınıf "1".
5	0,9999	0,3169	Eğer (F2=0) ve (F8=1) ve (F13=1) ise sınıf "1".
6	0,9999	0,3094	Eğer (F19=0) ve (F22=1) ise sınıf "1".
7	0,9999	0,3006	Eğer (F5=0) ve (F8=1) ise sınıf "1".
8	0,9912	0,2025	Eğer (F4=0) ve (F15=0) ve (F20=1) ise sınıf "1".
9	0,9989	0,19	Eğer (F5=0) ve (F7=1) ve (F15=0) ise sınıf "1".
10	0,9999	0,15	Eğer (F1=0) ve (F3=0) ve (F10=0) ise sınıf "1".
11	0,9979	0,125	Eğer (F6=0) ve (F8=0) ve (F9=0) ve (F11=0) ve (F12=1) ve (F16=0) ise sınıf "0".
12	0,9972	0,0925	Eğer (F1=1) ve (F4=0) ve (F15=0) ve (F17=0) ve (F19=1) ve (F22=0) ise sınıf "0".

Eğitim Doğruluğu:	100	Güvenilirlik:	97,84
Test Doğruluğu:	96,79	Destek:	98,93
ÇKA En İyi MSE değeri:	0,09	Genellik:	96,79

Ek-1.8. Vote veri kümesinde çıkarılan örnek veri kümesi

Kural No	Kalite	Uygunluk	Elde edilen örnek kural kümesi
1	0,9938	0,9314	Eğer (Physician-fee-freeze=evet) ise sınıf "republican".
2	0,9961	0,7851	Eğer (el-salvador-aid=evet) ve (synfuels-corporation-cutback=hayır) ve (crime=evet) ise sınıf "republican".
3	0,9214	0,7687	Eğer (adoption-of-the budget-resolution=hayır) ise sınıf "republican".
4	0,9999	0,7687	Eğer (adoption-of-the budget-resolution=evet) ise sınıf "democrat".
5	0,9999	0,7485	Eğer (el-salvador-aid=hayır) ise sınıf "democrat".
6	0,9998	0,6961	Eğer (education-spending=hayır) ve (export-administration-act-south-africa=evet) ise sınıf "democrat".
7	0,9999	0,661	Eğer (mx-missile=evet) ise sınıf "democrat".
8	0,9963	0,6353	Eğer (synfuels-corporation-cutback=hayır) ve (education-spending=evet) ve (superfund-right-to-sue=evet) ve (duty-free-exports=hayır) ise sınıf "republican".
9	0,9999	0,6074	Eğer (superfund-right-to-sue=hayır) ise sınıf "democrat".
10	0,9981	0,5422	Eğer anti-satellite-test-ban=evet) ve (export-administration-act-south-africa=evet) ise sınıf "democrat".
11	0,9999	0,3455	Eğer (synfuels-corporation-cutback=evet) ve (export-administration-act-south-africa=evet) ise sınıf "democrat".
12	0,9999	0,2813	Eğer (Physician-fee-freeze=hayır) ve (crime=evet) ise sınıf "democrat".
13	0,9999	0,2651	Eğer (religious-groups-in-schools=evet) ve (synfuels-corporation-cutback=evet) ise sınıf "democrat".

Eğitim Doğruluğu:	99,49	Güvenilirlik:	100
Test Doğruluğu:	100	Destek:	100
ÇKA En İyi MSE değeri:	0,00002	Genellik:	100

Ek-1.9. WBC veri kümesinde çıkarılan örnek veri kümesi

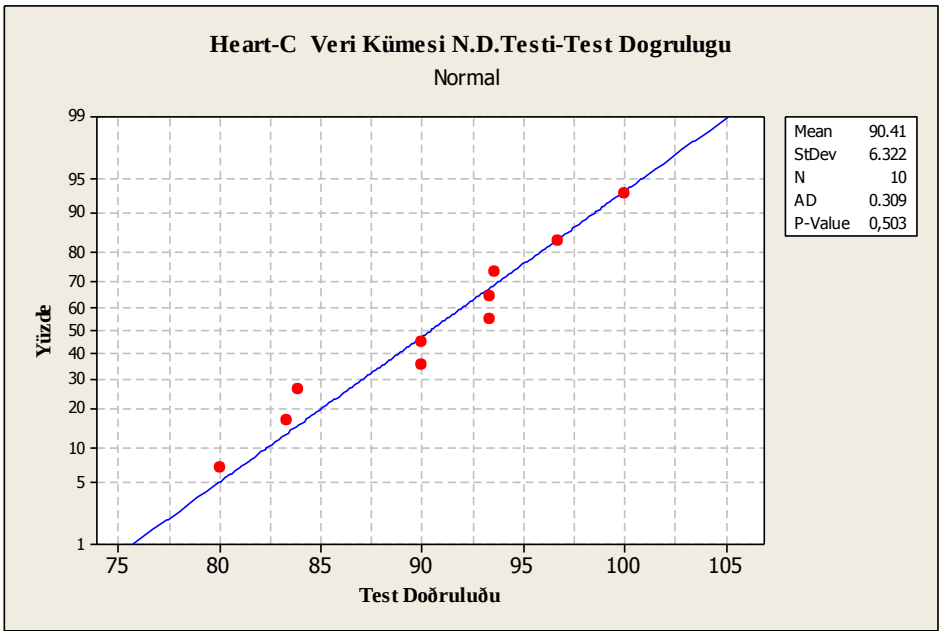
Kural No	Kalite	Uygunluk	Elde edilen örnek kural kümesi
1	0,9665	0,9018	Eğer (clump thickness<7) ve (uniformity of cell size<4 veya uniformity of cell size≥7) ve (marginal adhesion<4 veya marginal adhesion ≥7) ve (single epithelial cell size<4 veya single epithelial cell size≥7) ve (bare nuclei<7) ve (bland chromatin<7) ve (normal nucleoli<7) ve (mitoses<4) ise sınıf "2".
2	0,9993	0,9011	Eğer (clump thickness<7) ve (uniformity of cell size<4) ve (uniformity of cell shape<4 veya uniformity of cell shape≥7) ve (marginal adhesion<4) ve (bland chromatin<7) ve (normal nucleoli<7) ise sınıf "2".
3	0,9424	0,9002	Eğer (clump thickness<7) ve (uniformity of cell size<4) ve (uniformity of cell shape<4) ve (single epithelial cell size<4) ve (bland chromatin<4 veya bland chromatin≥7) ve (mitoses<4) ise sınıf "2".
4	0,9867	0,8886	Eğer (uniformity of cell size<4) ve (single epithelial cell size<7) ve (bare nuclei<4) ve (bland chromatin<4) ve (normal nucleoli<4) ve (mitoses<4) ise sınıf "2".
5	0,9075	0,8745	Eğer (clump thickness<7) ve (uniformity of cell size<4) ve (uniformity of cell shape<4) ve (bland chromatin<7) ve (mitoses<4) ise sınıf "2".
6	0,9995	0,857	Eğer (uniformity of cell size<4) ve (marginal adhesion<4) ve (bland chromatin<4) ve (normal nucleoli<7) ve (mitoses<4) ise sınıf "2".
7	0,9999	0,4347	Eğer (single epithelial cell size≥4) ve (bare nuclei<4 veya bare nuclei≥7) ve (mitoses<7) ise sınıf "4".
8	0,9999	0,4333	Eğer (bare nuclei<4 veya bare nuclei≥7) ve (normal nucleoli≥4) ve (mitoses<7) ise sınıf "4".
9	0,9999	0,399	Eğer (marginal adhesion<4 veya marginal adhesion≥7) ve (bare nuclei≥4) ve (bland chromatin<7) ve (mitoses<4) ise sınıf "4".
10	0,9999	0,3897	Eğer (clump thickness≥7) ve (single epithelial cell size<7) ise sınıf "4".
11	0,9999	0,3696	Eğer (clump thickness≥4) ve (uniformity of cell shapee≥4) ve (bland chromatin<4 veya bland chromatin≥7) ise sınıf "4".
12	0,9999	0,3668	Eğer (clump thickness≥4) ve (uniformity of cell shape ≥4) ve (single epithelial cell size≥4) ise sınıf "4".
13	0,9999	0,3551	Eğer (marginal adhesion<4 veya marginal adhesion≥7) ve

			(bland chromatin<7) ve (mitoses<4) ise sınıf "4".
14	0,9999	0,3501	Eğer (clump thickness≥4) ve (uniformity of cell size≥4) ve (single epithelial cell size≥4) ve (bare nuclei<4 veya bare nuclei≥7) ve (normal nucleoli<4 veya normal nucleoli≥7) ise sınıf "4".
15	0,9999	0,341	Eğer (clump thickness≥4) ve (uniformity of cell shape<7) ve (single epithelial cell size<7) ise sınıf "4".
16	0,9999	0,3061	Eğer (clump thickness≥4) ve (single epithelial cell size<7) ve (normal nucleoli<7) ise sınıf "4".
17	0,9999	0,2776	Eğer (single epithelial cell size<4 veya single epithelial cell size≥7) ve (bland chromatin≥4) ve (normal nucleoli<7) ise sınıf "4".
18	0,9999	0,2343	Eğer (uniformity of cell size<7) ve (bland chromatin<7) ve (normal nucleoli≥4) ise sınıf "4".
19	0,9999	0,1713	Eğer (clump thickness<7) ve (uniformity of cell size≥4) ve (single epithelial cell size<4 veya single epithelial cell size≥7) ve (normal nucleoli<4 veya normal nucleoli≥7) ise sınıf "4".
20	0,927	0,012	Eğer (uniformity of cell size<4 veya uniformity of cell size≥7) ve (4≤marginal adhesion<7) ve (bland chromatin<4 veya bland chromatin≥7) ve (normal nucleoli<7) ve (mitoses<4 veya mitoses≥7) ise sınıf "2".

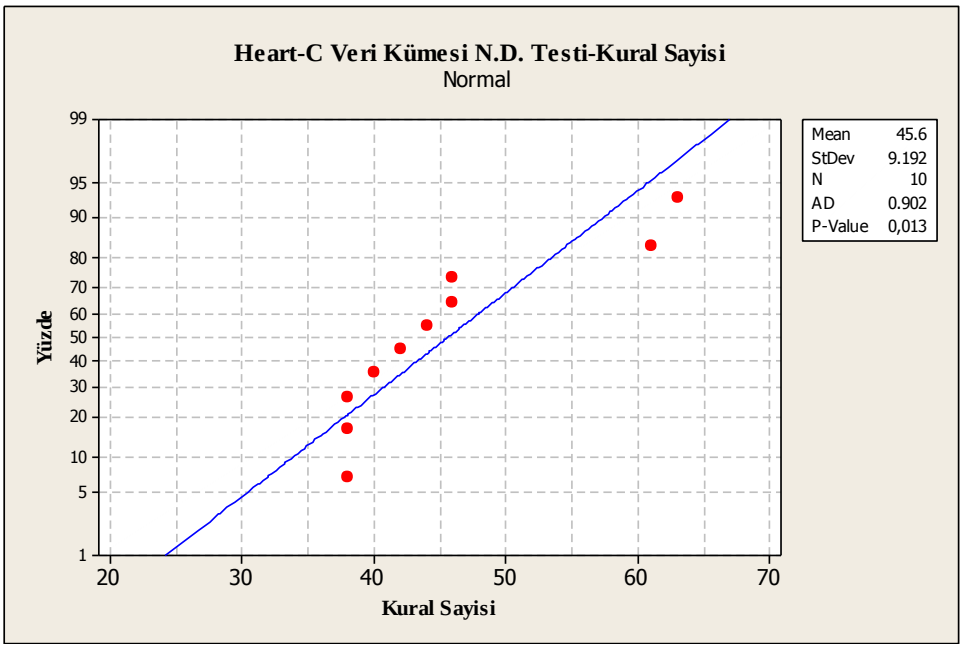
Eğitim Doğruluğu:	100	Güvenilirlik:	100
Test Doğruluğu:	100	Destek:	100
ÇKA En İyi MSE değeri:	0,03	Genellik:	100

EK-2 7. BÖLÜM NORMAL DAĞILIM TESTİ SONUÇLARI

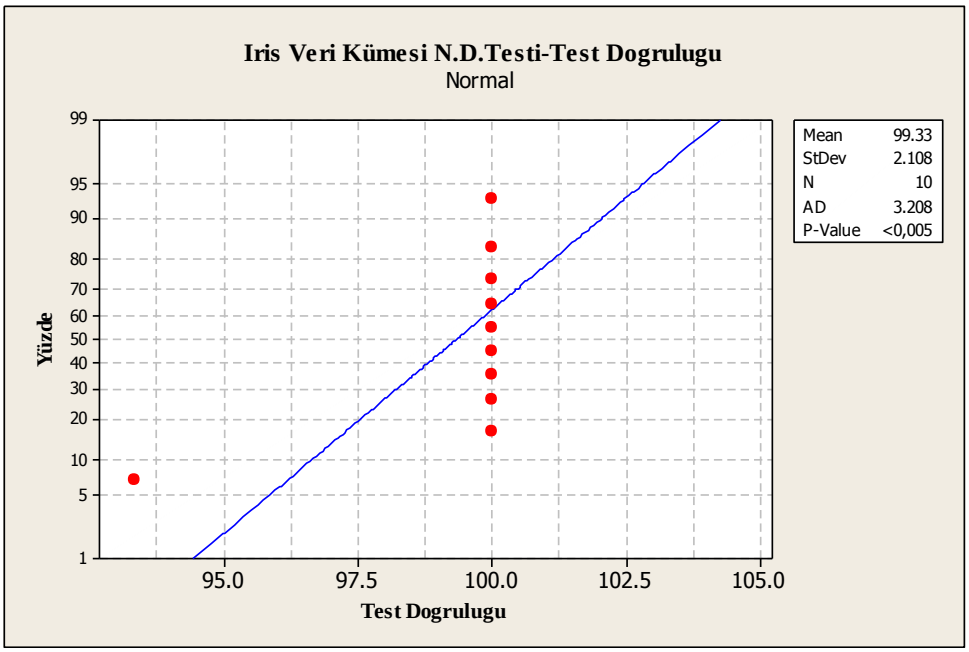
Ek-2.1a. Heart-C veri kümesi test doğruluğu normal dağılım testi



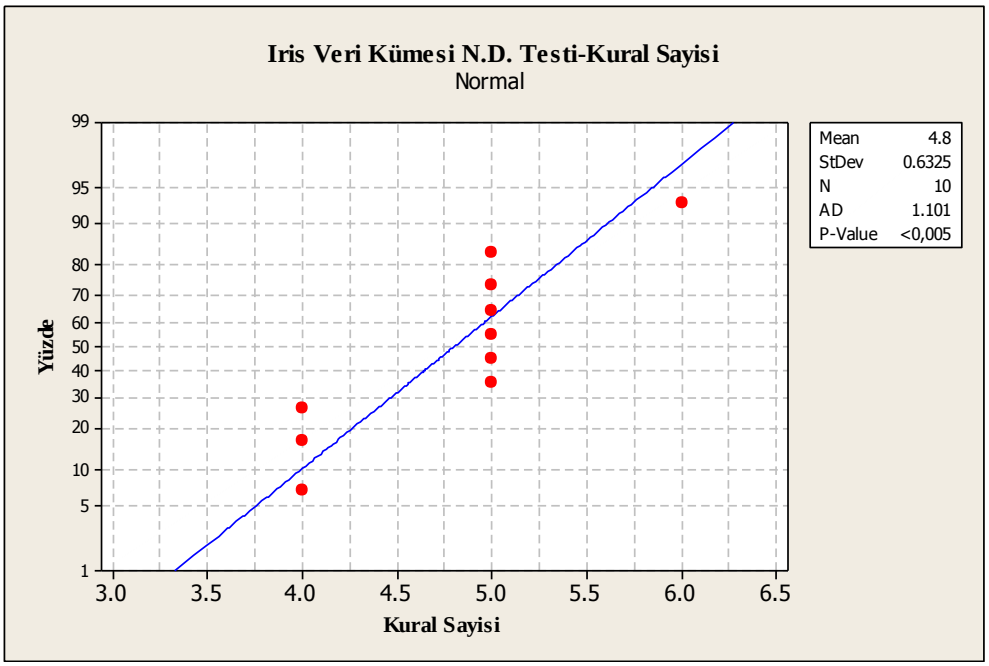
Ek-2.1b. Heart-C veri kümesi kural sayısı normal dağılım testi



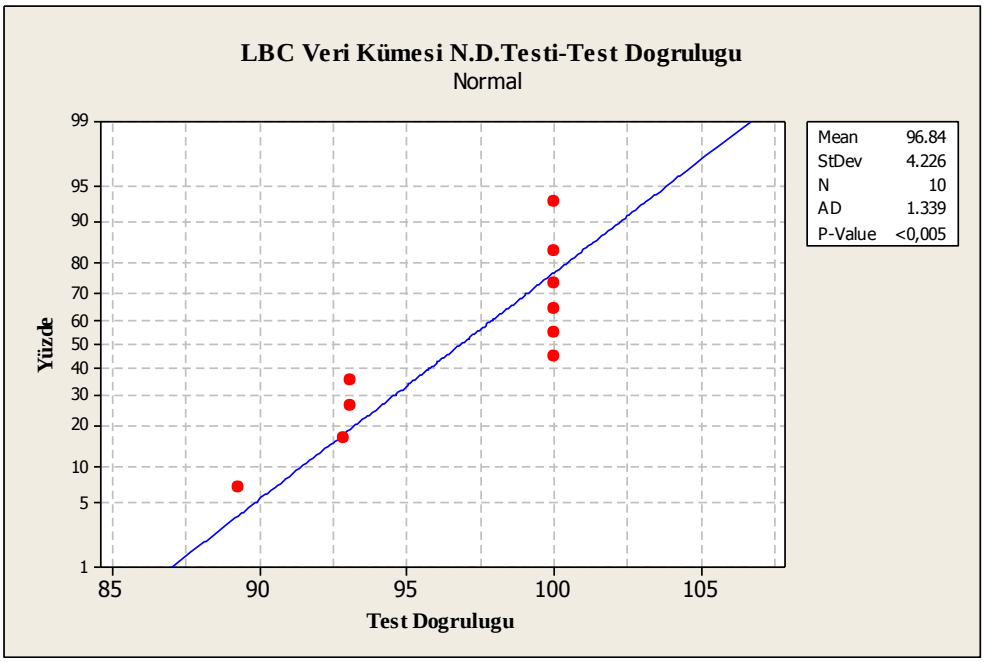
Ek-2.2a. Iris veri kümesi test doğruluğu normal dağılım testi



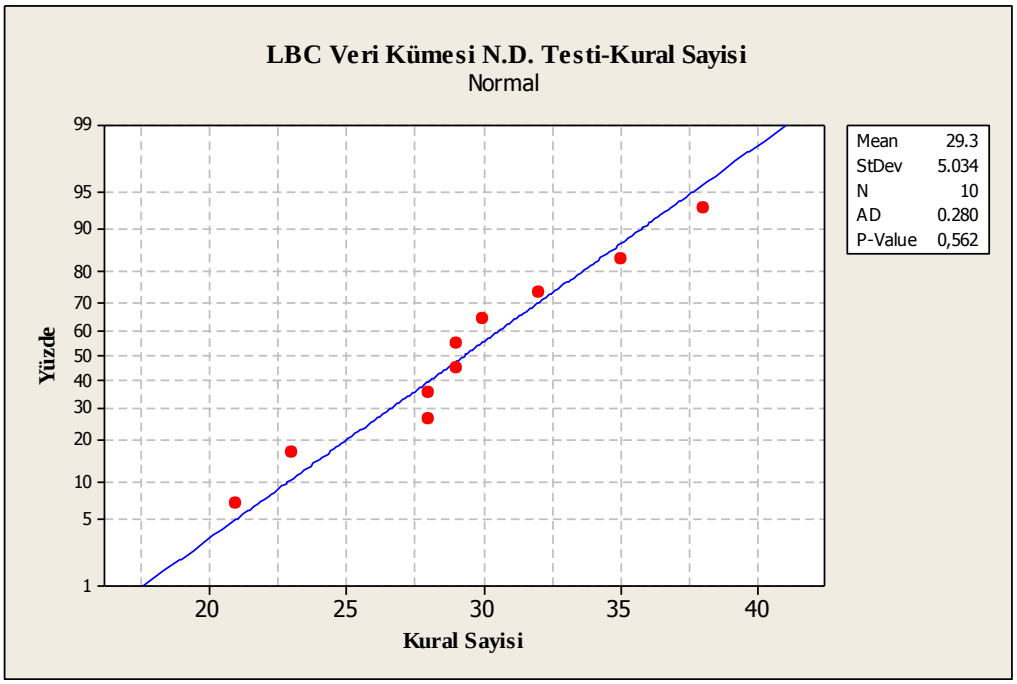
Ek-2.2b. Iris veri kümesi kural sayısı normal dağılım testi



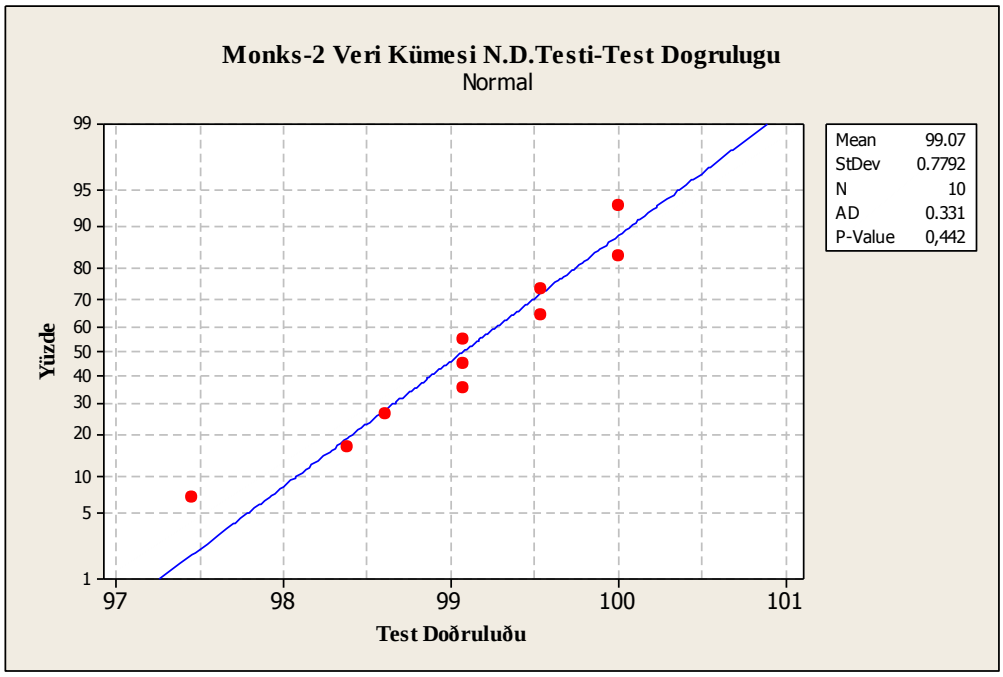
Ek-2.3a. LBC veri kümesi test doğruluğu normal dağılım testi



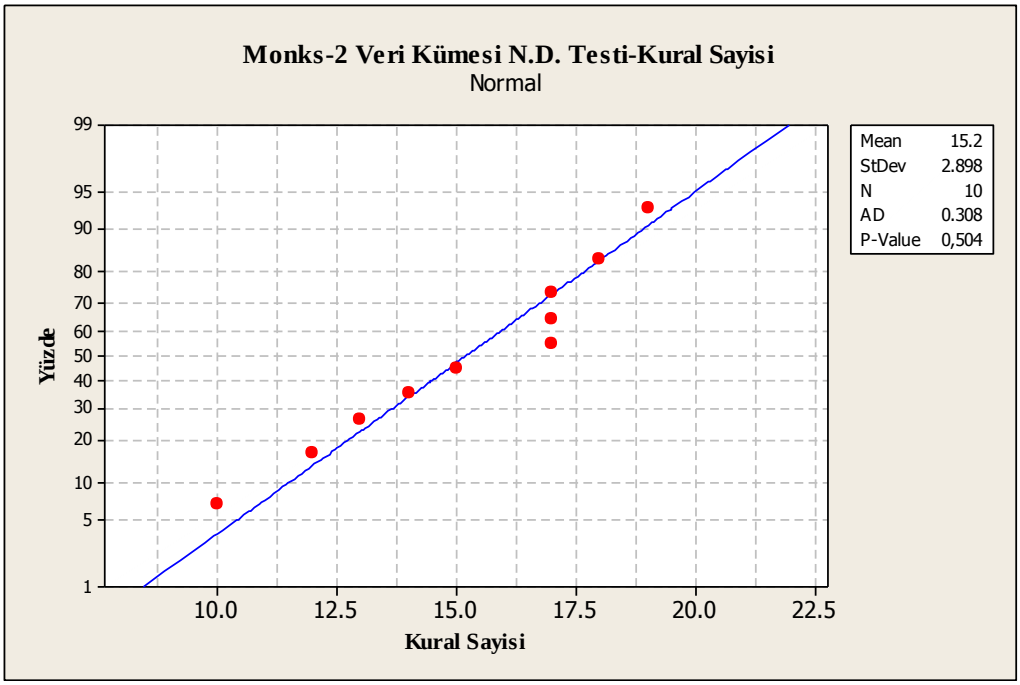
Ek-2.3b. LBC veri kümesi kural sayısı normal dağılım testi



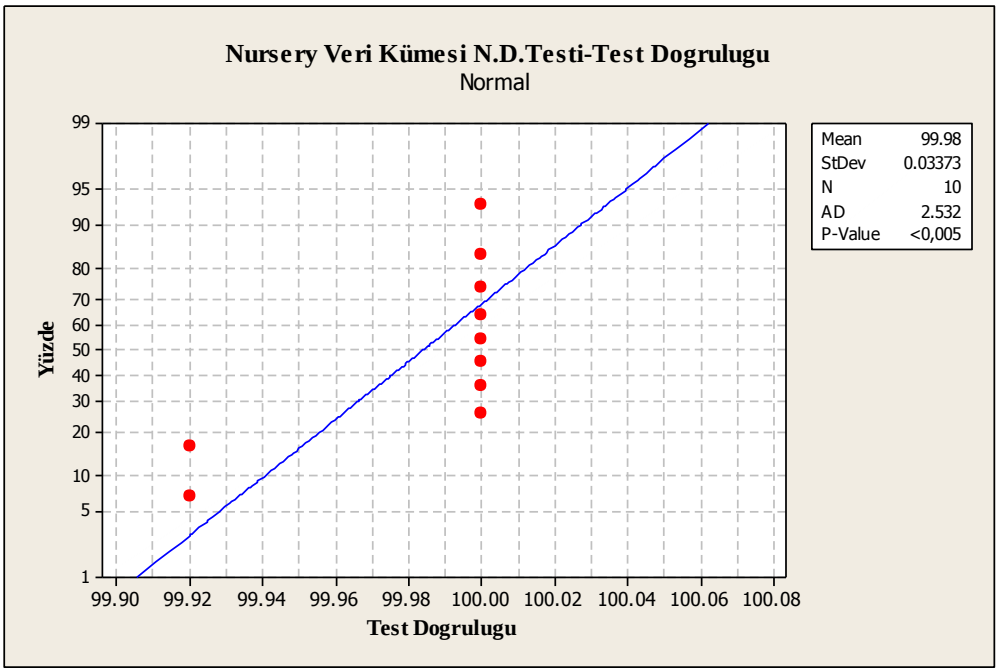
Ek-2.4a. Monks-2 veri kümesi test doğruluğu normal dağılım testi



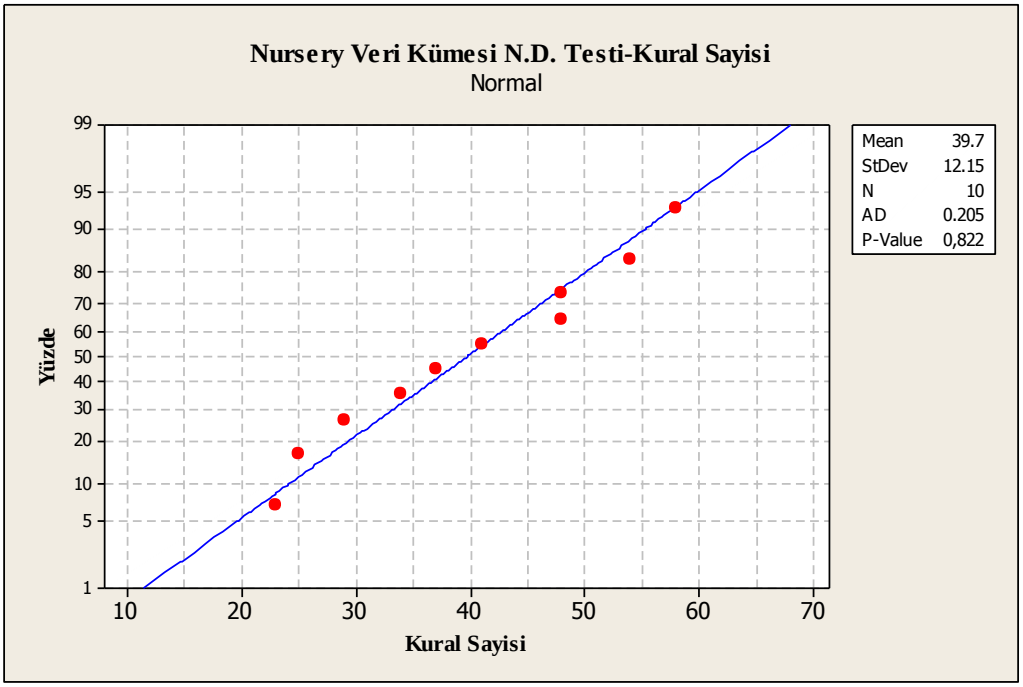
Ek-2.4b. Monks-2 veri kümesi kural sayısı normal dağılım testi



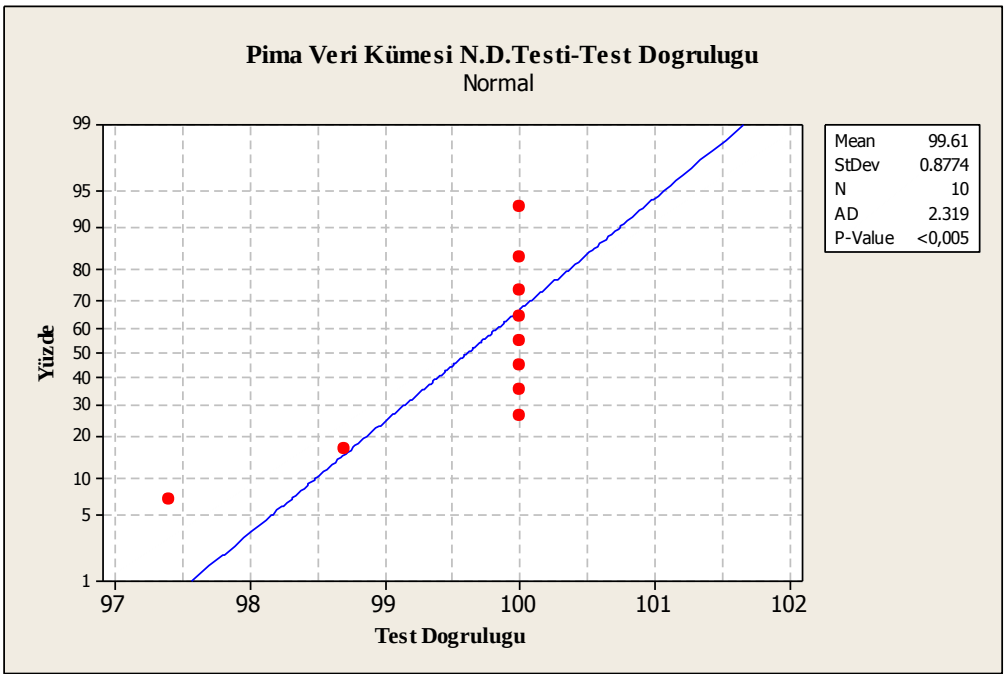
Ek-2.5a. Nursery veri kümesi test doğruluğu normal dağılım testi



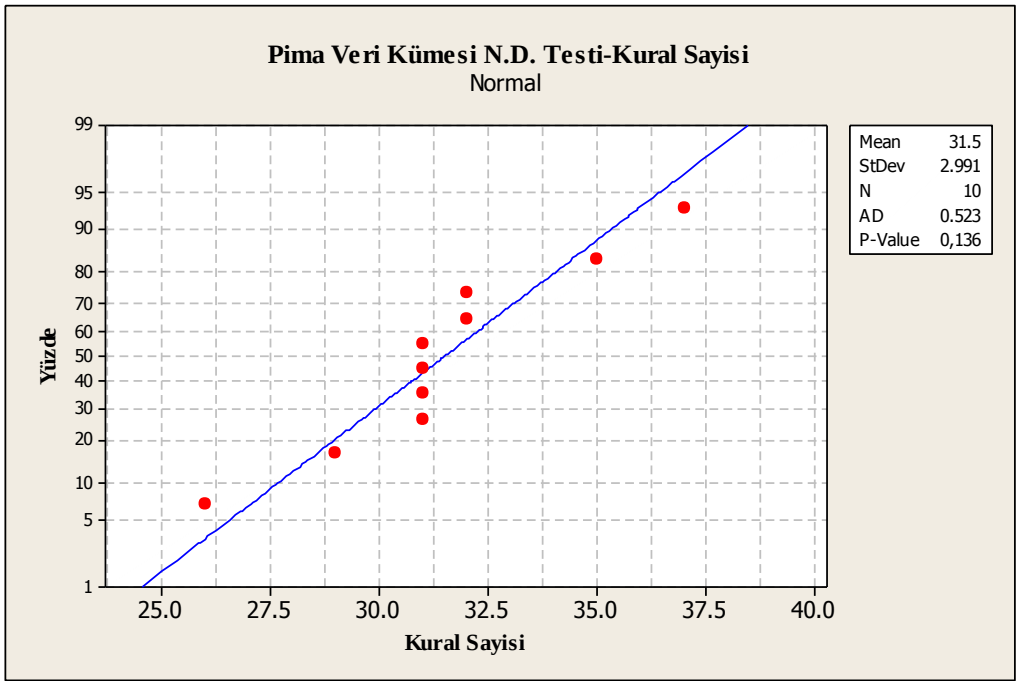
Ek-2.5b. Nursery veri kümesi kural sayısı normal dağılım testi



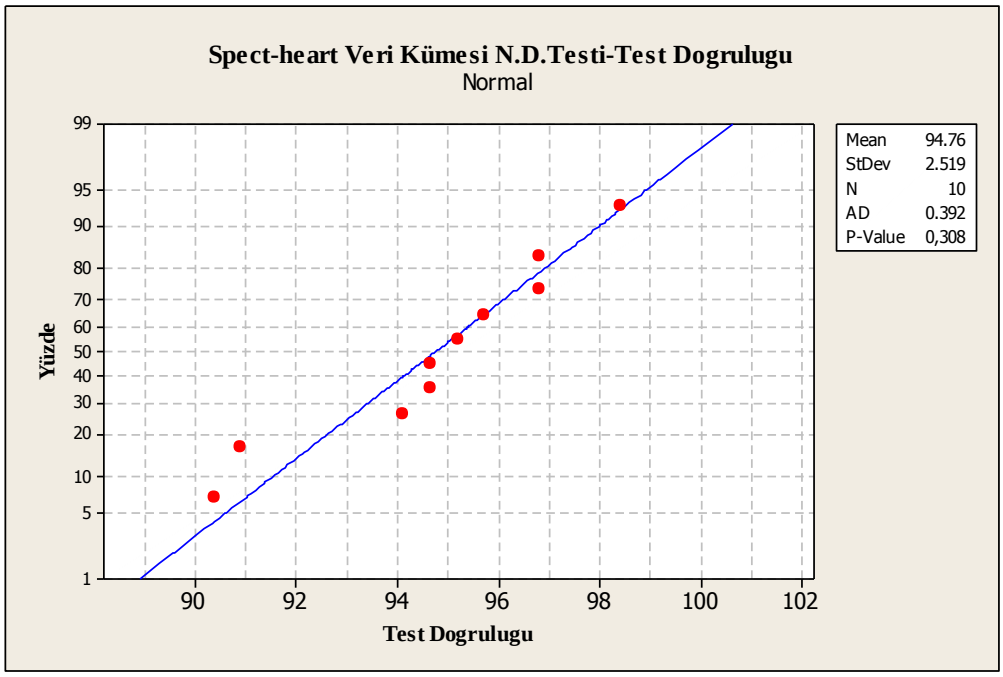
Ek-2.6a. Pima veri kümesi test doğruluğu normal dağılım testi



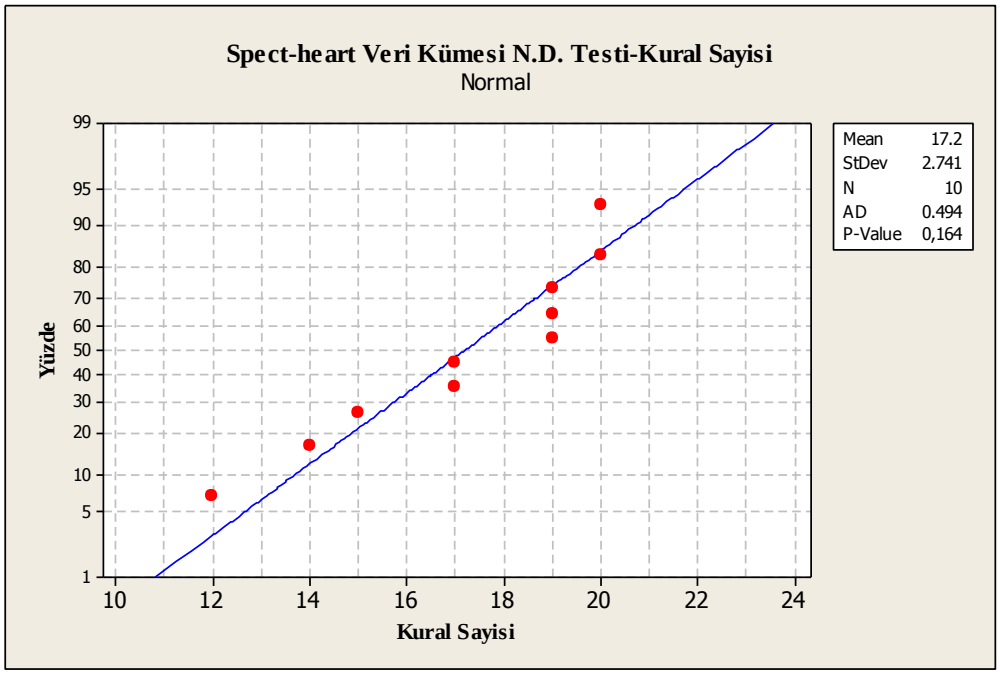
Ek-2.6b. Pima veri kümesi kural sayısı normal dağılım testi



Ek-2.7a. Spect-heart veri kümesi test doğruluğu normal dağılım testi



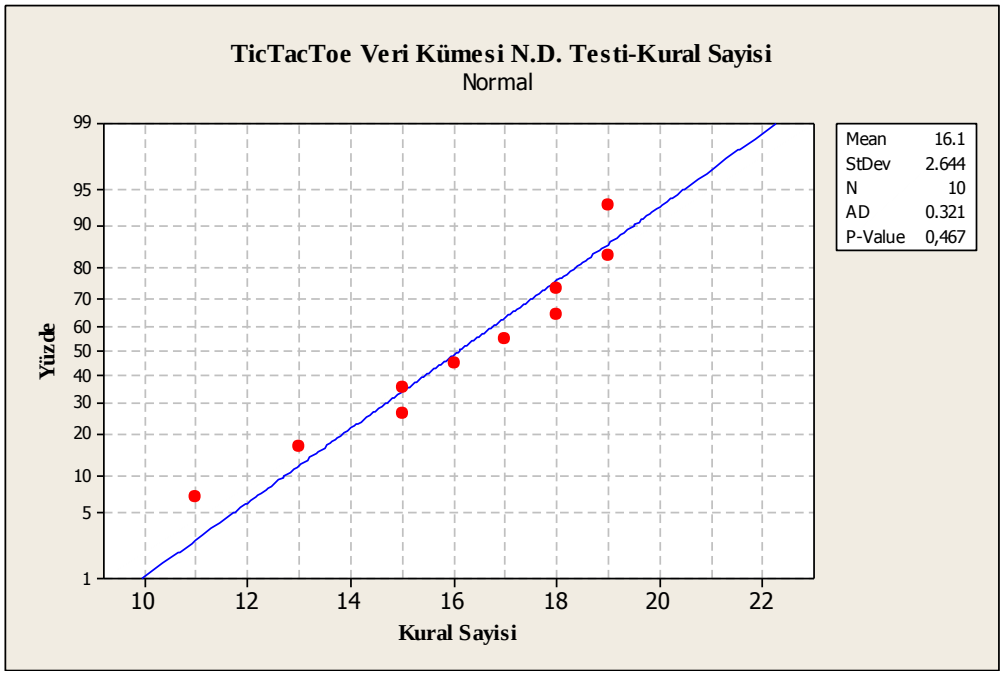
Ek-2.7b. Spect-heart veri kümesi kural sayısı normal dağılım testi



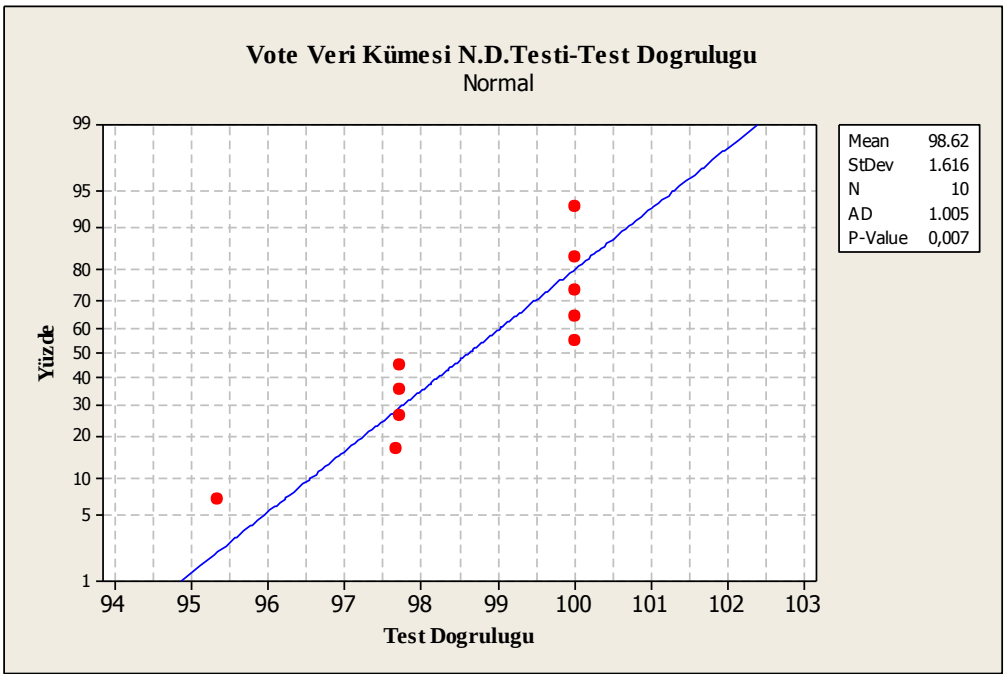
Ek-2.8a. Tic-Tac-Toe veri kümesi test doğruluğu normal dağılım testi

*** Geliştirilen TAKKOYSA-sınıflandırıcısı 10 örnekte de % 100 doğruluk elde ettiği ve standart sapma 0 olduğu için normal dağılım grafiği oluşturalamıştır.

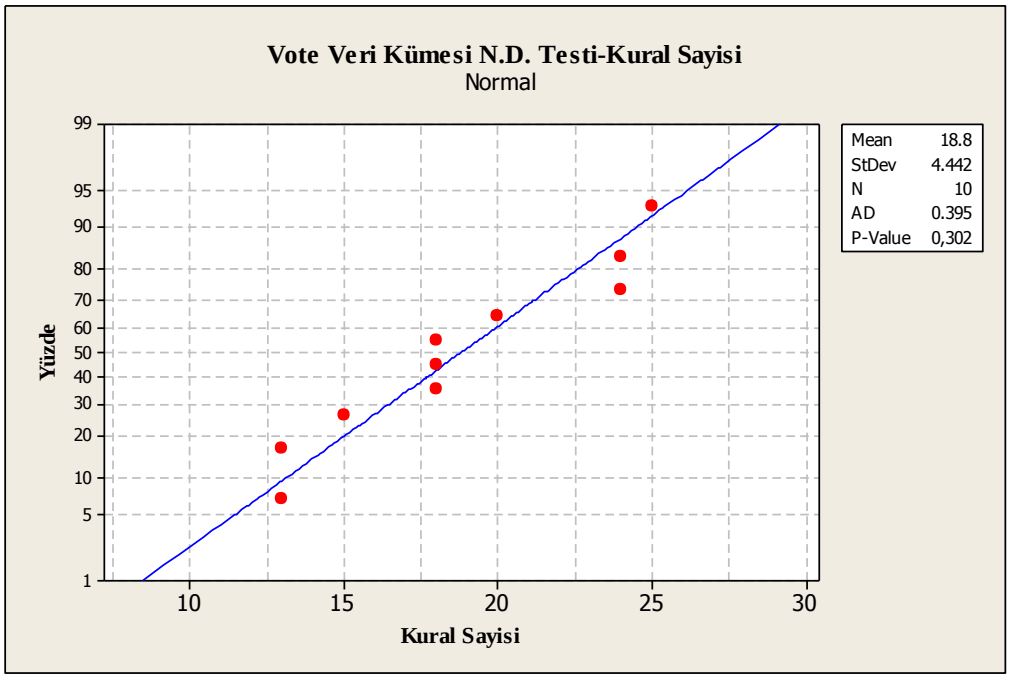
Ek-2.8b. Tic-Tac-Toe veri kümesi kural sayısı normal dağılım testi



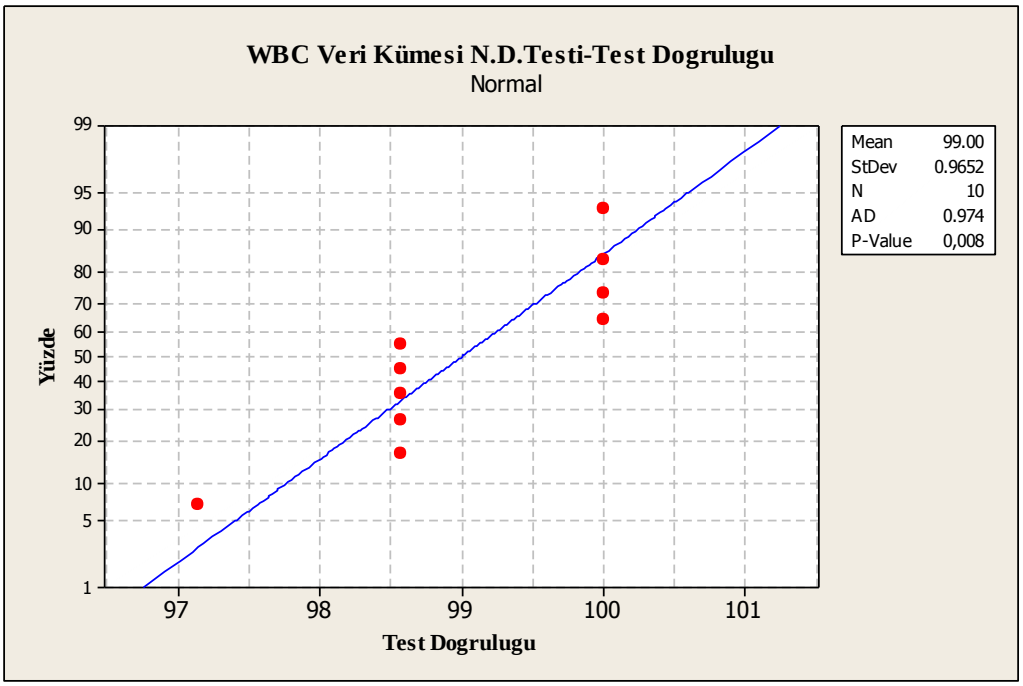
Ek-2.9a. Vote veri kümesi test doğruluğu normal dağılım testi



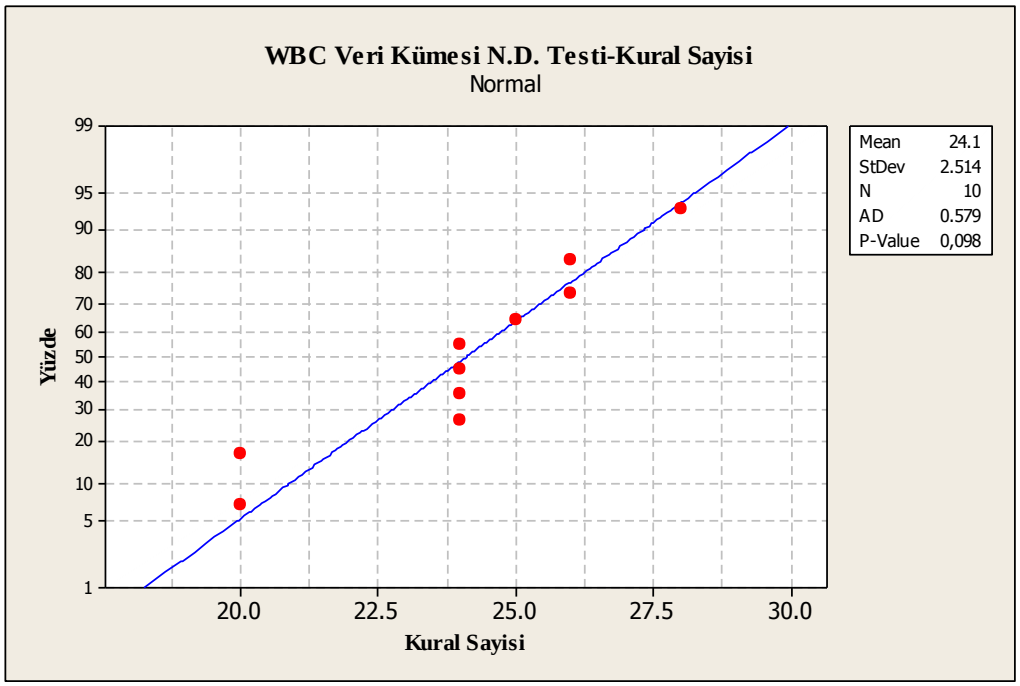
Ek-2.9b. Vote veri kümesi kural sayısı normal dağılım testi



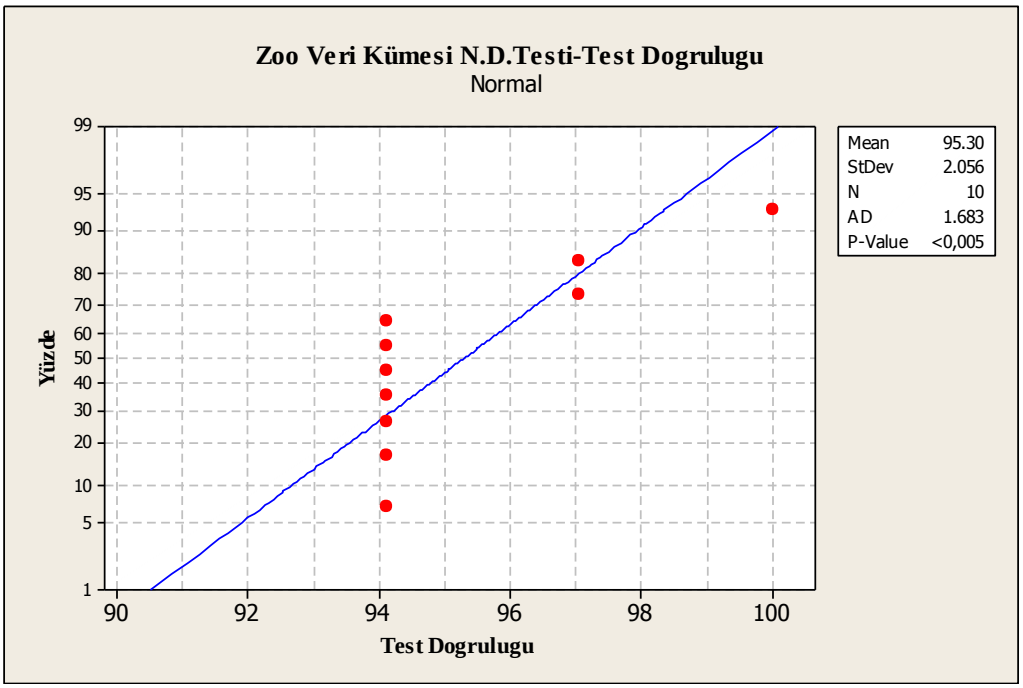
Ek-2.10a. WBC veri kümesi test doğruluğu normal dağılım testi



Ek-2.10b. WBC veri kümesi kural sayısı normal dağılım testi



Ek-2.11a. Zoo veri kümesi test doğruluğu normal dağılım testi



Ek-2.11b. Zoo veri kümesi kural sayısı normal dağılım testi