T.C.
İSTANBUL ÜNİVERSİTESİ
SOSYAL BİLİMLER ENSTİTÜSÜ
İŞLETME ANABİLİM DALI
SAYISAL YÖNTEMLER BİLİM DALI

YÜKSEK LİSANS TEZİ

Yapay Sinir Ağları
ve
Tahmin Modellemesi Üzerine Bir Uygulama

Fırat BAYIR

2501030188

Tez Danışmanı:

Prof. Dr. M. Erdal BALABAN

İstanbul, 2006

ÖZ

Bu çalışmada amaç, tahmin tekniklerinde kullanılan bir araç olarak yapay sinir ağlarının incelenmesi ve istatistiksel bir yöntem olan çoklu doğrusal regresyon analizi ile karşılaştırılmasıdır.

İlk bölümde yapay sinir ağları gelişim süreci ve fonksiyonel yapısı genel olarak anlatılmış, ikinci bölümde teorisi detaylı olarak incelenmiştir. Üçüncü bölümde sınıflandırılma şekilleri, dördüncü bölümde ise sık kullanılan ağ topolojileri hakkında detaylı bilgiler verilmiş Beşinci bölümde yapay sinir ağı tasarım aşamasından bahsedilmiştir.

Uygulamaya ayrılmış altıncı bölümde, Türkiye'deki imalat sanayi ihracat değerleri için yapay sinir ağları ve çoklu doğrusal regresyon analizi ile tahmin modelleri kurularak karşılaştırılmış ve sonuçlar yorumlanmıştır.

Anahtar Kelimeler: Yapay Sinir Ağları, Çoklu Doğrusal Regresyon Analizi, İmalat Sanayi İhracat Değerleri

ABSTRACT

The main purpose of this study is to investigate artificial neural networks as tool of forecasting techniques and to compare with statistical method, the multiple linear regression analysis.

In the first section, the evolution and the functional structure of artificial neural networks are introduced in general terms and detailed theory is examined in the second section. In the third section, the classifications of neural networks and in the forth section, phases of the designing of neural networks are mentioned. In the fifth section, the detailed knowledge of frequently used neural networks is given.

The sixth section is reserved to the comparison of two forecasting methods; artificial neural networks and multiple linear regression analysis by means of modeling Turkish Production Industry Exports values and results of both methods are compared and discussed.

Keywords: Artificial Neural Networks, Multiple Linear Regression Analysis, Turkish Production Industry Exports

ÖNSÖZ

Alan Turing'in kendi adını verdiği makinesinin icadıyla başlayan, insan oğlunun kendi kendine düşünen, karar verip harekete geçen makineler yapma çabası, yapay zeka kavramının ve paralelinde, insan beyninin öğrenme ve karar verme fonksiyonlarını taklit eden Yapay Sinir Ağlarının geliştirilmesi ile devam etmiştir. Yapay sinir ağlarının öğrenme ve genelleme yapabilme özellikleri, esneklik ve güçlü olmalarını sağlamakta ve onları bu yolla karar verme noktasında vazgeçilmez araçlar haline getirmektedir.

Yapay sinir ağları, doğrusal olmayan yapıları ve süreklilikleri sayesinde, tahmin, fonksiyon yaklaştırma, desen sınıflandırma, veri ilişkilendirme, kümeleme, veri filtreleme, optimizasyon ve kontrol amacıyla, finansal alanlardan, tıp alanına, savunma sanayinden, otomasyon ve kontrol alanlarına kadar hemen hemen tüm alanlarda sıkça kullanılan yapay sinir ağları geleneksel sistemlerden farklı çalışma yöntemleri ile doğrusal olmayan bir çok problemim çözümünde başarı ile kullanılmaktadır.

Bu çalışmada yapay sinir ağlarının teorisi ve kullanım alanları hakkında genel bilgiler verilmiş ve tahmin tekniği olarak kullanılmasına ilişkin bir uygulama yapılmıştır.

İlk bölümde yapay sinir ağlarının gelişimi, geleneksel sistemlerden farkları incelenmiş, üstünlükleri ve uygulama alanları hakkında bilgiler verilmiştir.

Çalışmanın ikinci bölümünde, yapay sinir hücrelerinin çalışma şekli, yapay sinir hücrelerinin bileşenleri ve ağın eğitimi detaylı olarak incelenmiştir.

Üçüncü bölümde, yapay sinir ağlarının sınıflandırılma şekilleri detaylı olarak anlatılmış, kullanım amaçlarına göre ağ topolojilerinden bahsedilmiştir.

Dördüncü bölümde ağ topolojileri hakkında bilgiler verilerek sık kullanılan bazı ağ topolojileri hakkında detaylı bilgiler verilmiştir.

Çalışmanın beşinci bölümünde yapay sinir ağı modeli kurulumu ve bileşenlerinin seçimi üzerinde durulmuştur.

Altıncı ve son bölümde Türkiye'deki imalat sanayi ihracat değerleri için çoklu doğrusal regresyon analizi ve yapay sinir ağları modelleri kurulmuş ve bu modellerin tahmin performansları karşılaştırılmıştır.

Bu çalışmada yardımlarını esirgemeyen, başta ailem olmak üzere, danışman hocam Prof. Dr. M. Erdal Balaban'a, yol gösterici olan Ergün Eroğlu'na, uygulamada benimle birlikte emek harcayan Neslihan Fidan'a, çalışmalarından ve fikirlerinden faydalandığım Nuray Baş'a, grafik tasarımda harikalar yaratan Burçin Özdeş'e ve tüm arkadaşlarıma teşekkür ederim.

İÇİNDEKİLER

ŞEKİL DİZİNİ

TABLO DİZİNİ

KISALTMALAR LİSTESİ

ABD:	Amerika Birleşik Devletleri
AE:	Mutlak Hata
ANN:	Yapay Sinir Ağları
ARE:	Bağıl Mutlak Hata
ART:	Uyarlamalı Rezonans Teorisi
EMB:	Eşik Mantık Birimi
GSYIH:	Gayri Safi Yurt İçi Hasıla
IMKB:	İstanbul Menkul Kıymetler Borsası
LM:	Lagrange Katsayısı
LMS:	En Küçük Kareler
LTM:	Uzun Dönemli Hafıza
LVQ:	Doğrusal Vektör Parçalama/Niceleme
MAE:	Ortalama Mutlak Hata
MAPE:	Ortalama Mutlak Yüzde Hata
ME:	Ortalama Hata
MLP:	Çok Katmanlı Algılayıcı
MPE:	Ortalama Yüzde Hata
MSE:	Ortalama Hata Kareleri
MSPE:	Ortalama Yüzde Hata Kareleri
PBNN:	Olasılık Tabanlı Yapay Sinir Ağları
RBF:	Radyal Tabanlı Fonksiyon
RMSE:	Ortalama Hata Kareleri Kökü
RMSPE:	Ortalama Yüzde Hata kareleri Kökü
SOM:	Özörgütlemeli Özellik Haritası
STM:	Kısa Dönemli Hafıza
SUI:	Sanayi Üretim Endeksi
TLU:	Eşik Mantık Birimi
USE:	Ulusal Sınai Endeks
VIF:	Varyans Artış Faktörü
VLSI:	Büyük Ölçekli Entegre Teknolojisi
YSA:	Yapar Sinir Ağları

GİRİŞ

Günümüzde YSA'lar bir çok alanda yaygın ve etkin bir şekilde kullanılmaktadır. Ekonomik tahmin de bu alanlardan bir tanesidir. Literatürde özellikle finansal değişkenlerin modellenmesi ve tahmini konusunda bir çok çalışma bulunmaktadır.

Bu çalışmanın amacı, bir tahmin tekniği olarak yapay sinir ağlarını tanıtmak ve istatistiksel bir yöntem olan çoklu doğrusal regresyon analizi ile karşılaştırmaktır.

Çalışma altı bölümden oluşmakta ve ilk bölümde yapay sinir ağlarından genel olarak bahsedilmektedir. Bu bağlamda yapay sinir ağlarının gelişimi, geleneksel sistemlerden farkları incelenmiş, üstünlükleri ve uygulama alanları hakkında bilgiler verilmiştir.

Çalışmanın ikinci bölümünde, öncelikle biyolojik sinir hücresinin yapısından bahsedilmiş ve ardından yapay sinir hücrelerinin çalışma şekli anlatılmıştır. Yapay sinir hücrelerinin bileşenleri tanıtılmış, ardından yapay sinir ağlarının eğitimi detaylı olarak incelenmiştir. Eğitim kapsamına giren öğrenme metotları, hata fonksiyonları, öğrenme katsayısı ve öğrenme kuralları bu bölüm dahilinde detaylı olarak tanıtılmıştır.

Üçüncü bölümde yapay sinir ağlarının sınıflandırılması üzerinde durulmuştur. Yapay sinir ağlarının öğrenme metotlarına göre, yapılarına göre ve uygulama alanlarına göre nasıl sınıflandırıldıkları, hangi amaçla hangi ağların kullanıldığı detaylı olarak incelenmiştir.

Dördüncü bölümde en basit ağlardan başlayarak çok katmanlı geri yayılım ağı anlatılmış, ardından yinelemeli ağlar ve özörgütlemeli ağlar hakkında bilgiler verilerek bu sınıfların öncüleri hakkında detaylı bilgiler verilmiştir.

Çalışmanın beşinci bölümünde bir yapay sinir ağı modeli kurarken nelere dikkat edilmesi gerektiği hususu üzerinde durulmuş, ağ topolojisinin seçimi, katman ve nöron sayısının belirlenmesi, aktivasyon, birleştirme ve performans fonksiyonlarının nasıl seçileceği konuları anlatılmış, performansı etkileyen faktörler üzerinde durulmuştur.

Altıncı ve son bölümde öncelikle uygulamada kullanılan veri setlerinin tanıtımı yapılmış, uygulamada kullanılacak modelin seçimi ve nedenleri anlatılmıştır. Çoklu doğrusal regresyon analizi uygulaması gerçekleştirilmiş, daha sonra yapay sinir ağları modelinin uygulamasına geçilmiştir. Türkiye'deki imalat sanayi ihracat değerleri için çoklu doğrusal regresyon analizi ve yapay sinir ağı modellerinin tahmin performansları karşılaştırılmıştır. Elde edilen sonuçlar doğrultusunda yorumlar yapılmış, yapay sinir ağları modelinin çoklu doğrusal regresyon analizi modeline göre üstün olduğu kanıtlanmıştır.

1. Bölüm: Yapay Sinir Ağları'na Giriş

Yüzyılın en büyük gelişmelerinden biri **Turing Makinesi**'nin icadıdır. Bu gelişme insanlar tarafından rutin olarak yapılan bazı işlemlerin makineler tarafından daha hızlı ve güvenilir yapılmasının önünü açmış, ardından bilgisayar teknolojisi gelişmiştir. Daha sonraki gelişmelerle bilgisayarlar hayatımızın içine yerleşmiş ve her alanda vazgeçilmez bir araç haline gelmiştir. Fakat insanoğlunun hayalleri bitmemiş, makinelerin insanın düşünebilme, öğrenebilme, yorumlayabilme ve karar verme yeteneklerini taklit edebilmesini sağlayan **Yapay Zeka** kavramının gelişmesi ile kendi kendine düşünebilen makinelerin önü açılmıştır.

Yapay Zeka araştırmaları Turing Makinesiyle başlamış, bir bilim dalı olarak kabul edilmesi Newel ve Simon tarafından sağlanmıştır. Yapay zeka, bilgisayar biliminin, insan beyninin otomasyonunu, çalışma şeklini, bilgi saklama ve işleme yeteneklerini araştıran ve kullanan dallarından biridir.

Son yıllarda yapay zeka konusunda yapılan araştırmalar, insan beyninin öğrenme ve karar verme fonksiyonlarını taklit eden **Yapay Sinir Ağları** (Artificial Neural Networks, ANN) üzerinde yoğunlaşmıştır.

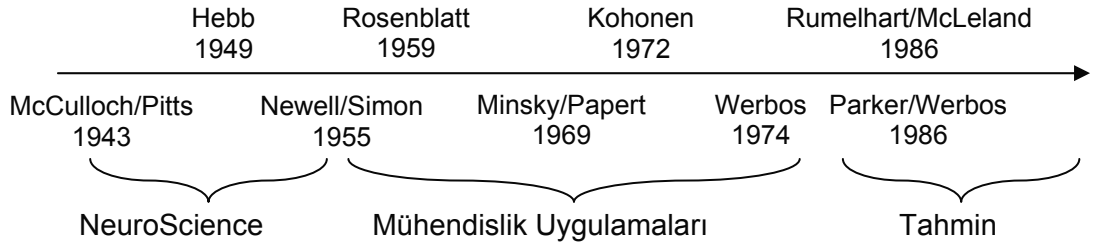
Yapay sinir ağlarının (YSA) öğrenme ve genelleme yapabilme özellikleri, esneklik ve güçlü olmalarını sağlamakta ve onları bu yolla karar verme noktasında vazgeçilmez araçlar haline getirmektedir. Bu çalışmada yapay sinir ağlarının çalışma sistemleri, türleri, öğrenme şekilleri ve kullanım alanları hakkında genel bilgiler verilmiş ve bir tahmin tekniği olarak kullanılmasına ilişkin bir uygulama yapılmıştır.

1.1. Yapay Sinir Ağları'nın Gelişimi

"Yapay sinir ağları, insan beynin sinir hücresinin çalışma şeklini taklit ederek, sistemlere öğrenme, genelleme, hatırlama özelliklerinin kazandırılması" (Saraç, 2004, s.9) şeklinde ifade edilir.

Bir diğer YSA tanımı, ilk ticari yapay sinir ağının geliştiricisi olan Dr. Robert Hecht-Nielsen'e aittir: "Yapay sinir ağı, dışarıdan gelen girdilere dinamik olarak yanıt oluşturma yoluyla bilgi işleyen, birbiriyle bağlantılı basit elemanlardan oluşan bilgiişlem sistemidir" (Caudill, 1987, s.47). Bu tanıma yakın bir tanım da YSA yazınında çok tanınan Teuvo Kohonen'e ait bir tanımdır: "Yapay sinir ağları, paralel bağlı çok sayıdaki basit elemanın, gerçek dünyanın nesneleriyle biyolojik sinir sisteminin benzeri yolla etkileşim kuran hiyerarşik bir organizasyondur" (Kohonen, 1987, s.1).

Yapay zeka konundaki ilk araştırmalar, Turing'in hesaplama modelini kullanarak yapay sinir ağlarının fizyolojisi ve önermeler mantığı konusunda çalışan McCulloch ve Pitts tarafından yapılmıştır. Fonksiyonları sadece "ve" "veya" mantıksal operatörlerini kullanarak modellemeye çalışmışlar, nöron mantığı ile yapay sinir hücrelerinin öğrenme yeteneklerini kazanabileceklerini göstermişlerdir. Hebb, Minsky, Edmonds, Mc Carthy, Shannon ve Rochester gibi bilim adamlarının araştırmaları sonraki YSA çalışmalarına öncülük etmiştir. Newell ve Simon ilk kuram ispatlayan programlarını tanıtmışlar ve ortaya attıkları **Fiziksel Simge Varsayımı**, insandan bağımsız zeka sistemleri ile uğraşanların hareket noktalarını oluşturmuştur (Saraç, 2004, s.5-6). Şekil 1.1'de YSA'ların gelişiminin tarihçesi gösterilmektedir.



Şekil 1.1 Yapay Sinir Ağlarının Gelişimi Tarihçe Zaman Çizelgesi (CRONE, s.12)

Minsky ve Papert'in 1969'da yayınladıkları "***Perceptron***" adlı kitaplarında tek katmanlı yapay sinir ağlarının XOR (Exclusive OR) problemi gibi çok basit problemleri çözemeyeceklerini göstermelerinden sonra YSA çalışmaları yavaşlamış, ancak Hoppfield, Amari, Anderson, Arbib, Fukushima, Grossberg, Kohonen, Little, Malsburg, Broomhead, Lowe, Parker ve Werbos gibi bilim adamlarının çalışmaları sayesinde yapay sinir ağları geliştirilmiş ve günümüzdeki şeklini almıştır. Rumelhart

ve McClelland'ın çok katmanlı ağlar için geliştirdikleri **Geri Yayılım Algoritmasını** (Back-Propagation) kullanan Parker ve Werbos'un çalışmaları ile 1986 senesinde XOR problemi çözülmüştür. 90'lı yıllardan itibaren YSA çalışmaları hız kazanmış, YSA'lar, çok çeşitli ve hızlı öğrenme algoritmalarının geliştirilmesi ile teorik ve laboratuar çalışmalarından çıkmış, günlük hayatta karşılaşılan problemlerin çözümünde yaygın olarak kullanılmaya başlanmıştır.

1.2. Yapay Sinir Ağları'nın Geleneksel Sistemlerden Farkları

Yapay sinir ağları, verilerin analizi ve veri içindeki desenin ortaya çıkarılması için geleneksel hesaplama yöntemlerinden farklı bir çözüm yöntemi sunarlar. Fakat tüm hesaplama problemlerine çözüm bulamazlar. Yapay sinir ağlarının da içinde bulunduğu **Uzman Sistemler** geleneksel hesaplama yöntemlerinin geliştirilmiş halidir ve 5. nesil hesaplama olarak anılmaktadır. İlk nesil hesaplama yöntemi Turing Makinesinin gelişimiyle ortaya çıkan kablolar ve anahtarlardan oluşan sistemlerdi. İkinci nesil sistemler transistorun gelişimi ile ortaya çıktılar. Üçüncü nesil, katı hal teknolojisinin gelişimini izleyen sistemlerdi. Dördüncü nesilde gelişmiş devreler, entegreler kullanılmaya başlandı. COBOL, FORTRAN, C gibi son kullanıcıya yönelik programlama dilleri geliştirildi. Dördüncü neslin şu anda geldiği nokta silikon teknolojisiyle üretilen hızlı işlemcilerdir. Beşinci nesil hesaplama yöntemi yapay zeka'yı da kapsamaktadır. Tablo 1.1 Geleneksel Hesaplama Yöntemleri ile Yapay Sinir Ağları Hesaplama Yöntemi arasındaki temel farkları göstermektedir.

Tablo 1.1: Geleneksel Hesaplama Yöntemleri ile YSA'ların Karşılaştırılması (Anderson ve McNeill, 1992, s.13)

Karakteristik	Geleneksel Hesaplama (Uzman Sistemler Dahil)	Yapay Sinir Ağları
İşlem Sitili	Sıralı	Paralel
Fonksiyonlar	Kurallar, Kavramlar ve Hesaplama Yoluyla, Mantıksal (Sol Beyin)	Resimler, Görüntüler, Kontroller Yoluyla, Geştalt (Sağ Beyin)
Öğrenme Metodu	Kurallarla (Didaktik)	Örneklerle (Sokratik)
Uygulamalar	Muhasebe, kelime işlem, matematik, stok, dijital iletişim	Sensor işleme, ses tanıma, desen ve desen tanıma, karakter tanıma, sınıflandırma

Tipik olarak uzman sistemler iki bölümden oluşmaktadır. **Çıkarsama Motoru** (Inference Engine) ve **Bilgi Deposu** (Knowledge Base). Çıkarsama motoru kullanıcı ara yüzü, dış dosyalar, program girişi, zamanlama, programlama ve çizelgeleme içerir. Bilgi deposu, belirli bir problem hakkında bilgiler içerir. Bu bilgi deposu kullanıcıya, yürüteceği işlem için gerekli bilgileri sağlar. Kullanıcı geleneksel hesap yönteminin nasıl çalıştığını bilmek zorunda değildir, sadece bilgisayarın yapması gerekenin ne olduğunu ve uzman sistem mekanizmasının nasıl çalışması gerektiğini bilmesi yeterlidir. Bu noktada çıkarsama motoru uzmanın isteklerini nasıl uygulayacağını iletmektedir. Uygulama; uzman sistemin kendisinin karar verip uygulamasıdır. Fakat uzman sistemin uygulayacağı işlemlerin kodlanması ve kuralların tanımlanması oldukça karmaşıktır. Sistem karmaşıklığı arttıkça çok fazla işlemci kaynağına ihtiyaç duyulmakta ve işlem süresi oldukça yavaşlamaktadır.

YSA'lar problem çözümü için tamamıyla farklı bir yaklaşım sunmakta ve altıncı nesil hesaplama olarak anılmaktadır. Programlamayı ve öğrenmeyi kendisi yapmaktadır. Yapay sinir ağları işlem sürekliliğinde bir uzmana ve programlamaya ihtiyaç duymadan çalışacak şekilde tasarlanmıştır. Veri içindeki, kimsenin orada olduğunu bilmediği deseni, örüntüyü tarar ve kendi başına öğrenir. Yapay Zeka Uzman Sistemleri ile Yapay Sinir Ağlarının karşılaştırılması Tablo 1.2'de belirtilmiştir.

Tablo 1.2 Uzman Sistemler ve YSA'ların Karşılaştırılması (Anderson ve McNeill., 1992, s.14)

Karakteristik	Uzman Sistemlerde Kullanılan Von Neumann Mimarisi	Yapay Sinir Ağları
İşlemci	Geleneksel İşlemciler	Yapay Sinir Ağları. Çok değişken teknolojilerde, yazılım ve donanım.
Hafıza	Ayrık	Tek ve aynı
İşlem yaklaşımı	Bir anda tek bir işlem ve tek bir kural, sıralı	Çoklu, eşzamanlı, paralel
Bağlantı	Dışarıdan programlama	Dinamik kendi kendine programlama
Kendi başına Öğrenme	Sadece algoritmik parametreler iyileştirildiğinde	Sürekli uyum sağlayan
Hata Toleransı	Özel işlemciler olmadığı takdirde yok	Bağlantılı olan nöronların yapısı gereği var
Programlama	Kavram ve kurallara bağlı, karmaşık	Kendi kendine programlanabilen, fakat ağın önceden tasarlanması gerekmekte

Uzman sistemler ciddi bir başarı kazanmıştır, fakat yapay zeka çalışmalarında görüş, ses tanıma ve sentezleme gibi konularda sorunlar yaşanmıştır. Yapay zeka, yazılımın çalıştığı işlemciye bağımlı kalmak zorunda ve tek bir işlemcinin limitleriyle sınırlanmaktadır.

Yapay sinir ağlarının uzman sistemler ve geleneksel hesaplama yöntemleri karşısındaki avantajlarına karşın bir çok problem için tam bir çözüm olamamaktadır. Öğrendikleri kadar hata yapmaya da devam etmektedirler. Üstelik bir ağ geliştirildikten sonra bu ağın optimal bir ağ olduğunu öğrenmenin bir yolu yoktur. Yapay sinir ağları kendi istediği gibi davranır ve uygulayıcısından bazı gereksinimleri sağlamasını bekler. Bu gereksinimler:

- Problemi karakterize edebilecek bilgiyi içeren veri setine ihtiyaç duyar,
- Eğitim ve test aşamaları için yeterli büyüklükte, ayrık veri setlerine ihtiyaç duyar,
- Problemin iyi anlaşılmasına ihtiyaç duyar. Problem iyi anlaşıldığı takdirde ağın yapısı doğru kurulabilir ve ağın kullanacağı topoloji, fonksiyonları ve öğrenme kuralları doğru bir şekilde belirlenebilir.

Bu gereksinimler sağlandıktan sonra yapay sinir ağları, geleneksel işlemcilerin zayıf işlemci gücü ve adım adım çalışma sistemiyle çözemeyeceği problemlerin çözümü için olanak sağlar. Bir çok karmaşık problem geleneksel hesaplama yöntemleri ile çözülememektedir. Konuşma, sağlam olan herkesin kolayca işleyip anlayabileceği bir şeydir. İnsanlar yaşadıkları coğrafyanın dilinin farklı ağız ve lehçelerini, bir bebeğin çıkardığı sesleri anlayabilir. Yapay sinir ağlarının paralel işlemci gücü haricinde bu işlemlerin bir bilgisayar tarafından yapılması imkansızdır. Görüntü algılama ve desen tanıma yine bir insanın kolayca yapabileceği ama çok büyük bilgisayarların bile yapamayacağı işlerdendir. Bir insan hiç görmediği bir el yazısını okuyup anlayabilir veya bir uçağın bulutların arasından çıkışını, dönüşünü, üzerinden uçuşunu ve nokta olup kayboluncaya kadar uzaklaşıp küçülmesini izleyip algılayabilir. Geleneksel bir bilgisayar ise değişen görüntüleri veri tabanındaki milyonlarca farklı resimle karşılaştırmaya çalışacaktır.

1.3. Yapay Sinir Ağlarının Üstünlükleri

Bölüm 1.2'de bahsedildiği gibi yapay sinir ağları özellikle doğrusal olmayan sistemlerde, geleneksel sistemlere göre oldukça fazla üstünlüklere sahiptir. Öngörü ve tahmin gibi konularda ve öğrenme noktasında sahip oldukları süreklilik sayesinde istatistik tekniklere göre daha kolay çözüm üretebilirler. Bundan dolayı başta işletmecilik ve finans olmak üzere bir çok değişik alanda kullanım imkanı bulurlar. YSA'ların üstünlükleri şu şekilde özetlenebilmektedir.

Doğrusal Olmama: Yapay sinir ağlarının en önemli özelliklerinden birisi gerçek hayattaki olası doğrusal olmayan yapıları da dikkate alabilmesidir. YSA' nın temel işlem elemanı olan hücre, doğrusal değildir. Dolayısıyla hücrelerin birleşmesinden meydana gelen YSA da doğrusal değildir ve bu özellik bütün ağa yayılmış durumdadır. YSA'lar bu özellikleri ile doğrusal olmayan karmaşık problemlerin çözümünde önemli araçlardan biri haline gelmiştir.

Paralellik: Bölüm 1.2'de bahsedildiği gibi günümüzde kullanılan bilgi işleme yöntemleri genelde seri, sıralı işlemlerden oluşmaktadır. Seri işlemcilerde herhangi bir birimin yavaş olması tüm sistemi yavaşlatırken, YSA'ların sahip olduğu paralellik sayesinde yavaş bir birim sistemin çalışması sırasında herhangi bir soruna yol açmamaktadır. Bu durum YSA'ların daha hızlı ve güvenilir olması sonuçlarını doğurmaktadır.

Öğrenme: Geleneksel hesaplama yöntemlerinde bir problemin çözülebilmesi için probleme uygun bir algoritma geliştirilmesi ve programlama yolu ile hesaplama yapılması gerekmektedir. Genellikle bu tür algoritmaların çözüm yeteneği uzmanın kod yazma yeteneği ile sınırlıdır. Bu tür algoritmaların zorluğu ve her problem türüne göre farklı algoritma yazılma ihtiyacı nedeniyle karmaşık problemlerin çözümünde kullanılamazlar. Herhangi bir nesneyi bilgisayara geleneksel yöntemlerle öğretmek için nesnenin mümkün olan tüm açılardaki ve uzaklıklardaki görüntülerini, mümkün olan tüm değişik kombinasyonları ile birlikte göstermeniz gerekmektedir. YSA'ların öğrenme sistemi ise insan beyninin çalışma şekline benzemektedir. Öğrenme, özellikleri verilen örnekler yoluyla yapay sinir ağının kendisi tarafından sağlanmakta ve YSA'lar, örnekleri kullanarak probleme ilişkin genelleme yapabilecek yeteneğe ulaşmaktadır. Bu özelliği sayesinde geleneksel yöntemler için karmaşık olan

sorunlara çözüm üretilebilmektedir. Ayrıca, insanların kolayca yapabildiği ancak geleneksel yöntemler için imkansız olan basit işlemler için de uygun olmaktadırlar. Geleneksel sistemlerden ayrıldığı bir başka nokta ise sürekli öğrenmedir. YSA'lar, kendisine gösterilen yeni örnekleri öğrenebilmeleri ve yeni durumlara adapte olabilmeleri sayesinde sürekli olarak yeni olayları öğrenebilmesi mümkündür.

Bilginin Saklanması: Geleneksel hesaplama yöntemlerinde bilgi, veri tabanlarında veya program kodlarının içinde saklanmaktadır. Yapay sinir ağlarında ise bilgi, nöronlar arasındaki ağırlıklı bağlantılarda saklanmaktadır. Yani bilgi ağa dağıtılmış durumdadır ve ağın bütünü, öğrendiği olayın tamamını göstermektedir. Bu nedenle yapay sinir ağlarının dağıtılmış bellekte bilgi saklayabildikleri söylenebilmektedir.

Hata Toleransı ve Esneklik: Yapay sinir ağları, geleneksel işlemcilerden farklı şekilde işlem yapmaktadırlar. Geleneksel işlemcilerde tek bir merkezi işlemci eleman her hareketi sırasıyla gerçekleştirmektedir. Seri bilgi işlem yapan geleneksel bir sistemde herhangi bir birimin hatalı çalışması, hatta tamamen bozulmuş olması tüm sistemin hatalı çalışmasına veya bozulmasına sebep olacaktır. YSA modelleri, her biri büyük bir problemin bir parçası ile ilgilenen çok sayıda basit işlemci elemanlardan oluşmaları ve bağlantı ağırlıklarının ayarlanabilmesi gibi özelliklerinden dolayı önemli derecede esnek bir yapıya sahiptirler. Bu esnek yapı sayesinde ağın bir kısmının zarar görmesi modelde sadece performans düşüklüğü yaratmakta, problemin çözümünde büyük bir soruna yol açmamakta, modelin işlevini tamamen yitirmesi söz konusu olmamaktadır. Bu nedenle, geleneksel yöntemlere göre hatayı tolere etme yetenekleri son derece yüksektir. Ayrıca toplam işlem yükünü paylaşan işlemci elemanların birbirleri arasındaki yoğun bağlantı yapısı sinirsel hesaplamanın temel güç kaynağıdır. Bu yerel işlem yapısı sayesinde, yapay sinir ağları yöntemi en karmaşık problemlere bile uygulanabilmekte ve tatminkar çözümler sağlayabilmektedir (Yurtoğlu, 2005, s.35).

Genelleme: YSA'lar, kendi kendine öğrenme yeteneği sayesinde bilinen örnekleri kullanarak daha önce karşılaşılmamış durumlar için genelleme yapabilmektedirler. Yani, hatalı (gürültülü) veya kayıp veriler için çözüm üretebilmektedir. YSA'lar, daha önce görmedikleri veriler, veya eksik veriler hakkında karar verirken genelleme

yapabildikleri için iyi birer **Desen (Örüntü) Tanımlayıcısı** (Pattern Recognition Engine) ve **Sağlam Sınıflandırıcılardır** (Robust Classifier) (Yurtoğlu, 2005, s.36).

Uyarlanabilirlik: Yapay sinir ağları, ilgilendiği problemdeki değişikliklere göre ağırlıklarını ayarlayabilmektedir. Yani belirli bir problemi çözmek amacıyla eğitilen bir yapay sinir ağı, problemdeki değişimlere göre tekrar eğitilebilmekte, değişimler devamlı ise gerçek zamanda da eğitime devam edilebilmektedir. Bu özelliği ile yapay sinir ağları, uyarlamalı (adaptive) örnek tanıma, sinyal işleme, sistem tanılama ve denetim gibi alanlarda etkin olarak kullanılabilmektedir (Saraç, 2004, s.14).

Eksik Verilerle Çalışma: Yapay sinir ağları, geleneksel sistemlerin aksine, kendileri eğitildikten sonra eksik bilgiler ile çalışabilmekte ve gelen yeni örneklerde eksik bilgi olmasına rağmen sonuç üretebilmektedirler. Yapay sinir ağlarının eksik bilgiler ile çalışması performanslarının düşeceği anlamına gelmemektedir. Performansın düşmesi eksik olan bilginin önemine bağlı olmaktadır. Hangi bilginin önemli olduğunu ağın kendisi eğitim sırasında öğrenmektedir. Kullanıcıların bu konuda bir fikri yoktur. Ağın performansı düşük olunca, kayıp olan bilginin önemli olduğu, performans düşmez ise eksik bilginin önemli olmadığı sonucuna varılmaktadır.

Sınırsız Sayıda Değişken ve Parametre Kullanma: Yapay sinir ağları istatistiki yöntemlerin aksine sınırsız sayıda değişken ve parametre ile herhangi bir ekstra dönüşüme ihtiyaç duymadan çalışabilmektedir. Bu sayede mükemmel bir tahmin doğruluğu ile genel çözümler sağlanabilmektedir (Yurtoğlu, 2005, s.36).

Gerçeklenme Kolaylığı: Yapay sinir ağlarının basit işlemler gerçekleyen türden hücrelerden oluşması ve bağlantıların düzgün olması, ağların gerçeklenmesi açısından büyük kolaylık sağlamaktadır. Yapay sinir ağlarının farklı uygulama alanlarındaki yapıları da standart yapıdaki bu hücrelerden oluşacaktır. Bu nedenle, farklı uygulama alanlarında kullanılan yapay sinir ağları benzer öğrenme algoritmalarını ve teorilerini paylaşabilirler. Bu özellik, problemlerin yapay sinir ağları ile çözümünde önemli bir kolaylık getirmektedir (Saraç, 2004, s.13).

Donanım ve Hız: Yapay sinir ağları paralel yapısı nedeniyle **Büyük Ölçekli Entegre Devre Teknolojisi** (VLSI) ile gerçeklenebilir. Bu özellik, YSA'ların hızlı bilgi işleme yeteneğini arttırmaktadır ve gerçek zamanlı uygulamalarda yaygın olarak kullanılmasının ana sebeplerindendir.

Yapay sinir ağlarının yukarıda belirtilen avantajları geniş uygulama alanları bulmalarını sağlamaktadır. Ancak yapay sinir ağlarının göz önünde bulundurulması gereken bazı dezavantajları da bulunmaktadır. Bunlar arasında en önemlisi, geniş veri seti gereksinimidir. Yapay sinir ağları eğitilebilmesine ve test edilebilmesine yetecek genişlikte veri setine ihtiyaç duyulmaktadır. Fakat yeterli veri seti genişliği için kesin bir kriter yoktur, bir noktada uygulamaya bağlıdır.

Dezavantaj sayılabilecek diğer bir nokta ise basit olarak görülebilecek modelleme yapılarına rağmen bazı konularda uygulamanın zor ve karmaşık olabilmesidir. Bazı durumlarda, bir yakınsama sağlamak bile imkansız olabilmektedir fakat bu durum da uygulama alanına bağlıdır ve genellikle çok karmaşık problemlerde ortaya çıkmaktadır (Yurtoğlu, 2005, s.37). Doğru modelleme için genellikle deneme yanılma kullanılması da önemli bir dezavantaj sayılabilir. Çünkü kullanılan modelin doğruluğunu ve oluşturulan çözümün optimum çözüm olup olmadığını test etmek zordur. Model doğru kurulmuş bile olsa yapay sinir ağları optimum çözüm garantisi vermez, yalnızca kabul edilebilir çözümler üretebilir (Öztemel, 2003, s.34). Bu bir nokta da dezavantaj gibi görülse de, bir çok problemde, tam sonucun bulunamadığı veya optimum çözümün çok fazla zaman alacağı durumlarda en hızlı ve en uygun çözümü ürettiklerinden avantaj olarak değerlendirilebilir. Çünkü geçek dünyada çözümü karmaşık ve uzun zaman alacak problemlerle sık karşılaşılır ve rekabet şartları düşünüldüğünde hızlı ve iyi çözümler üretmek önemlidir. Bir çok ağ günümüzde istatistiksel olarak yüksek doğruluklarla (%85-%90 doğrulukla) çalışsa da bir çok kullanıcı problemlerinin tam çözümünü beklemektedir (Anderson ve McNeill, 1992, s.61).

Ağın öğreneceği veri setinin ağa gösterilmesi de başlı başına bir problemdir. Yapay sinir ağları sayısal (nümerik) veri haricinde veri kabul etmezler. Sayısal olmayan verilerin sayısal verilere dönüştürülmesi gerekmektedir. Bu ise tamamen kullanıcının becerisine bağlıdır (Öztemel, 2003, s.34). Sayısal olmayan verilerin ağa gösterilmesi

sorunu hala tam olarak çözülememiştir. Günümüzde bir çok olayın yapay sinir ağları yardımıyla hala çözülememiş olmasının en temel nedenlerinden biri budur.

Belki de en önemli sorun ağın davranışlarının açıklanamamasıdır. Bir çözüm üretildiği zaman neden ve nasıl bu çözüm üretildiği konusunda bilgi bulmak mümkün değildir (Öztemel, 2003, s.35). Bu yüzden yapay sinir ağlarına **Kara Kutu** (Black Box) adı verilmiştir. Bazı bilim adamları da yapay sinir ağlarını **Voodoo Mühendisliği** (Voodoo Engineering) olarak ifade etmektedir (Anderson ve McNeill, 1992, s.61). İçeride ne olup bittiğini kimse tam olarak bilemez. Bu ise ağa olan güveni azaltmaktadır.

1.4. Uygulama Alanları

Yapay sinir ağları eksik bilgilerle çalışabilme ve normal olmayan verilere çözüm üretebilme yeteneklerinden dolayı pek çok alanda kullanılmaktadırlar. Doğrusal olmayan, çok değişkenli problem uzayı olan, gürültülü, değişkenler arasında karmaşık etkileşimleri olan, kesin olmayan, eksik, kusurlu, hata olasılığı yüksek veriler ve problemlerin çözümü için özellikle matematiksel bir model ve algoritmanın bulunmaması durumlarında yaygın halde başarılı YSA uygulamaları yapılabilmektedir. Bu amaçla geliştirilmiş ağlar genel olarak şu fonksiyonları yerine getirmektedir:

- Probabilistik fonksiyon kestirimleri yaklaştırmaları,
- Sınıflandırma,
- Veri İlişkilendirme,
- Desen eşleştirme, tanıma,
- Kavramlaştırma / kümeleme,
- Zaman serileri analizleri,
- Sinyal filtreleme,
- Veri sıkıştırma,
- Doğrusal olmayan sinyal işleme,
- Doğrusal olmayan sistem modelleme,
- Optimizasyon,
- Zeki ve doğrusal olmayan kontrol (Öztemel, 2003, s.36).

Yapay sinir ağları bu teorik uygulamaların ötesinde günlük hayatta kullanılan finansal konulardan mühendisliğe ve tıp bilimine kadar bir çok alanda uygulanabilmektedir. Evimizdeki aletlerden cep telefonlarına kadar günlük hayatımızda YSA uygulamaları görmek mümkündür.

Bu uygulamaların hepsini bu çalışmada göstermek imkansızdır. Bu nedenle belli başlı uygulamalardan örneklerle bahsedilecektir. Bu uygulamalar aşağıdaki başlıklar altında özetlenebilir (Saraç, 2004, s.11) (Öztemel, 2003, s.205) (Stader, 1992, s.40-41) (Anderson ve McNeill, 1992, s. 61-67) (Jain, Mao ve Mohiuddin, 1996, s.8).

- Arıza analizi ve tespiti
- Finansal uygulamalar
- Tıp uygulamaları
- Savunma sanayi uygulamaları
- Haberleşme uygulamaları
- Üretim uygulamaları
- Otomasyon ve kontrol uygulamaları

Arıza Analizi ve Tespiti: Bir sistemin, cihazın ya da elemanın düzenli (doğru) çalışma şeklini öğrenen bir yapay sinir ağı yardımıyla bu sistemlerde meydana gelebilecek arızaların gerçek zamanlı tanımlanma olanağı olmaktadır. Bu amaçla YSA'lar, elektrik makinelerinin, uçakların ya da bileşenlerinin, entegre devrelerin v.s. arıza analizinde, mekanik parçaların ömürlerinin ve kırılmalarının tahmininde kullanılmaktadırlar.

Finansal Uygulamalar: Makro ekonomik tahminler, banka kredilerinin ve sigorta poliçelerinin değerlendirilmesi, kredi kartı hilelerinin tespiti, kredi kartı kurumlarında iflas tahminleri, emlak kredilerinin yönetilmesi, bond, hisse senedi ve döviz kuru tahminleri, risk analizleri gibi örneklerde uygulama alanı bulmaktadır.

Tıp Uygulamaları: EEG ve ECG gibi tıbbi sinyallerin analizi, kanserli hücrelerin analizi, protez tasarımı, transplantasyon zamanlarının optimizasyonu, kan hücreleri reaksiyonları ve kan analizlerinin sınıflandırılması, kalp krizlerinin önceden tespiti,

görüntüleme cihazlarının ürettiği verilerden hastalıkların teşhisi ve hastanelerde giderlerin optimizasyonu gibi konularda uygulanmaktadır.

Savunma Sanayi Uygulamaları: Silahların otomasyonunda ve hedef izlemede, radar ve sonar sinyallerinin sınıflandırılması ile nesneleri/görüntüleri ayırma ve tanımada, yeni sensor ve algılayıcı tasarımında, mayın detektörlerinde, uçakların rotalarının belirlenmesinde ve gürültü önleme gibi alanlarda kullanılmaktadır.

Haberleşme Uygulamaları: Görüntü ve veri sıkıştırma, otomatik bilgi sunma servisleri, iletişim kanallarındaki gereksiz yankılanmaların ve gürültülerin filtrelenmesi, iletişim kanallarındaki trafik yoğunluğunun kontrol edilmesi ve anahtarlanması gibi alanlarda.

Endüstriyel ve Üretim Uygulamaları: Endüstriyel bir süreçte fırınların ürettiği gaz miktarlarının tahmini, kimyasal süreçlerin dinamik modellenmesi, üretim sistemlerinin planlama ve çizelgeleme optimizasyonu, ürün analizi ve tasarımı, ürünlerin kalite analizi ve kontrolü, planlama, müşteri/talep tahmini ve ürünün pazardaki performansının tahmini ve yönetim analizi gibi alanlarda kullanılmaktadır.

Otomasyon ve Kontrol Uygulamaları: Uçaklarda otomatik pilot sistemi otomasyonu, zeki ulaşım araçlarında ve robotlarda otomatik optimum rota belirleme, robotik sistemlerin ve hareket mekanizmalarının kontrolü, doğrusal olmayan sistem modelleme ve kontrolü, elektrikli sürücü sistemlerin kontrolü gibi alanlarda ve insansız araçların tüm kontrollerinde uygulanmaktadır.

Diğer uygulamalar: Bunların yanında YSA'lar veri madenciliği, beyin fonksiyonlarının modellenmesi, ses ile çalışan teknolojik aletlerin geliştirilmesi, konuşmaların farklı dillere eş zamanlı tercümeleri (Simultaneous Translation) ve konuşmaları yazıya dönüştürmede (Speech Recognition) kullanılmaktadır. Günümüzde kullanılan sistemler konuşmalarda geçen 20.000 (İngilizce) kelimeyi algılayabilmekte ve yazıya geçirebilmektedir. Optik Karakter Tanıma (Optical Character Recognition) ve güvenlik sistemlerinde, konuşma ve parmak izi okuma alanlarında kullanılmaktadır. Kredi kart formlarının veya postalar üzerindeki adres ve posta kodlarının el yazısından tanınması gibi işlemlerde %98-%99 gibi yüksek

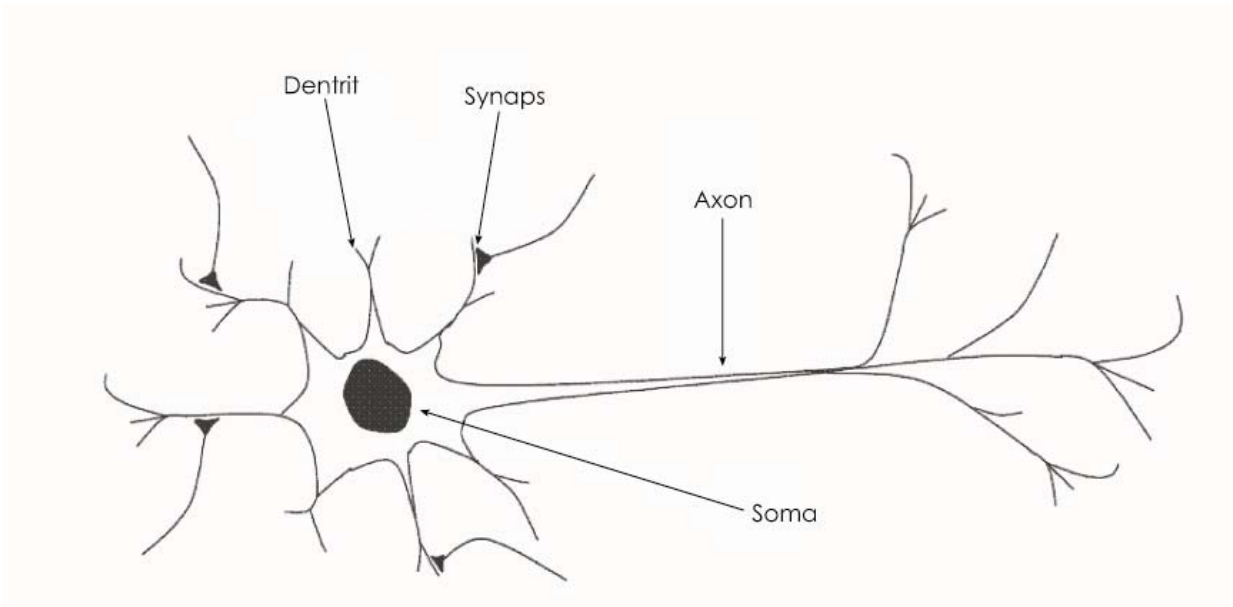
doğruluklarla kullanılmaktadır. Yine bankalarda imza ve el yazısı tanıma gibi güvenlik konularında kullanılmaktadır. Ayrıca, genetik mühendisliğinde DNA ve kromozomların incelenmesinde, gezgin satıcı problemi, yük ve hat dengeleme gibi optimizasyon problemlerinin çözümünde, petrol ve gaz aramalarında, hava alanlarında bomba detektörlerinde, akıllı oyuncakların geliştirilmesinde, büyük inşaat projelerinde, proje yönetiminde iş çizelgeleme ve maliyet hesaplarında, kimyasal süreçlerde hata tespiti gibi konularda kullanılmaktadırlar.

2. Bölüm: Yapay Sinir Ağları'nın Yapısı

2.1. Biyolojik Sinir Sistemi

Biyolojik Sinir Sistemi; sürekli olarak duyu organları ve diğer sinirlerden gelen bilgileri yorumlayarak uygun kararlar veren **Beyin** tarafından yönetilir. Beyin merkez olmak üzere üç katmandan oluşur. Bu katmanlar; çevreden gelen sinyalleri (girdileri) elektriksel sinyallere çeviren **Alıcı Sinirler** (Receptor) ve beynin ürettiği sinyalleri uygun tepkilere dönüştüren **Tepki Sinirleri** ve tepki sinirleri arasında sürekli ileri geri besleme yaparak uygun sinyaller üreten **Merkezi Sinir Sistemidir** (Saraç, 2004, s.18).

Sinir sisteminin temel elemanları **Nöron** adı verilen özel sinir hücreleridir. Bir sinir hücresi **Dentrit**'ler (Dendrite), **Hücre Gövdesi** (Soma), **Akson**'lar (Axon) ve **Sinaps**'lardan (Synapse) oluşur. Şekil 2.1 sinir hücresinin şeklini göstermektedir.



Şekil 2.1 Şekil Biyolojik Sinir Hücresinin Yapısı (Saraç, 2004, s.20)

Sinir hücreleri birbirlerine dentritlerinden veya aksonlarından bağlanır. Bu bağlantı noktalarına sinaps denir. Sinapslar bir diğer sinir hücresinin aksonundan aldıkları pozitif ya da negatif yönde elektrik akımının elektro kimyasal bir yöntemle dentrite iletilmesini sağlarlar.

Dentrit, aldığı elektrik akımını somaya yani hücrenin gövdesine iletir. Soma, çekirdeği barındırır ve hücrenin yaşamasını sağlar. Hücrenin tüm dentritlerinden alınan elektrik sinyalleri burada toplanır. Eğer hücrenin **Eşik Değeri** aşılmışsa, aksonlarda bir elektrik akımı oluşturulur ve bu elektrik akımı bir diğer hücrenin dentritine sinapslar üzerinden geçirilir.

Bir insanda ortalama 10^{11} sinir hücresi bulunduğu göz önünde bulundurulursa, sonsuz sayıda sinapsın bulunduğu ortaya çıkar. Sinir hücrelerinin iletim hızları bilgisayarlarınkinden daha yavaş olsa da sayısız sinapsın oluşturduğu bağlantılar, aynı anda birden fazla karar verilmesini ve beynin duyusal verileri çok hızlı değerlendirmesini sağlamaktadır. Bu açıdan insan beyninin karmaşık yapısı hala tam olarak çözülememiş ve bir çok fonksiyon hala tam olarak açıklanamamıştır. Tablo 2.1'de insan beyni ile bilgisayarların sayısal değerleri karşılaştırılmıştır.

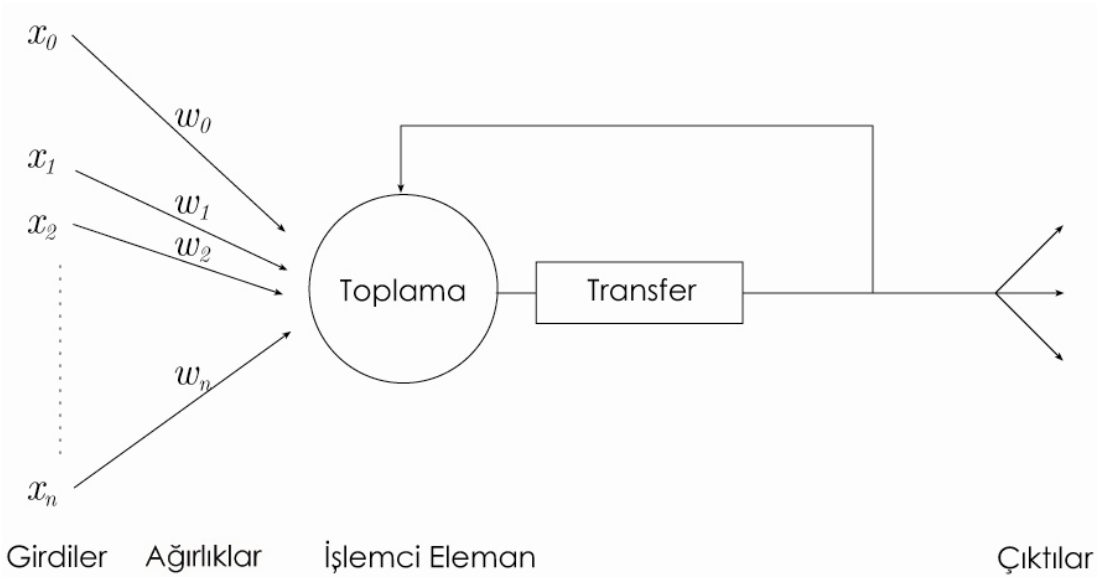
Tablo 2.1 İnsan Beyni ile Bilgisayarların Sayısal Olarak Karşılaştırılması (CRONE, s.13)

	İnsan Beyni	**Bilgisayar**
İşlemci Hızı	10^{-3} ms (0,25 MHz)	10^{-9} ms (2500 MHz)
Nöron / Transistör	10^{11} & 10^{33} bağlantı	10^{9} (Yonga)
Ağırlık	1500 g.	Kilogramlarca
Enerji Tüketimi	10^{-16} Joule	10^{-6} Joule
Hesaplama	100 Adım	Milyarlarca Adım

Biyolojik sistemlerde öğrenme, nöronlar arasındaki sinaptik bağlantıların ayarlanması ile oluşur. İnsanlar doğumlarından itibaren yaşayarak öğrenme sürecine girerler. Yaşadıkça beyin sürekli gelişmekte, yeni sinapslar oluşmakta, var olan sinapsların bağlantıları ve hücrelerin eşik değerleri ayarlanmaktadır. Öğrenme bu şekilde olmaktadır.

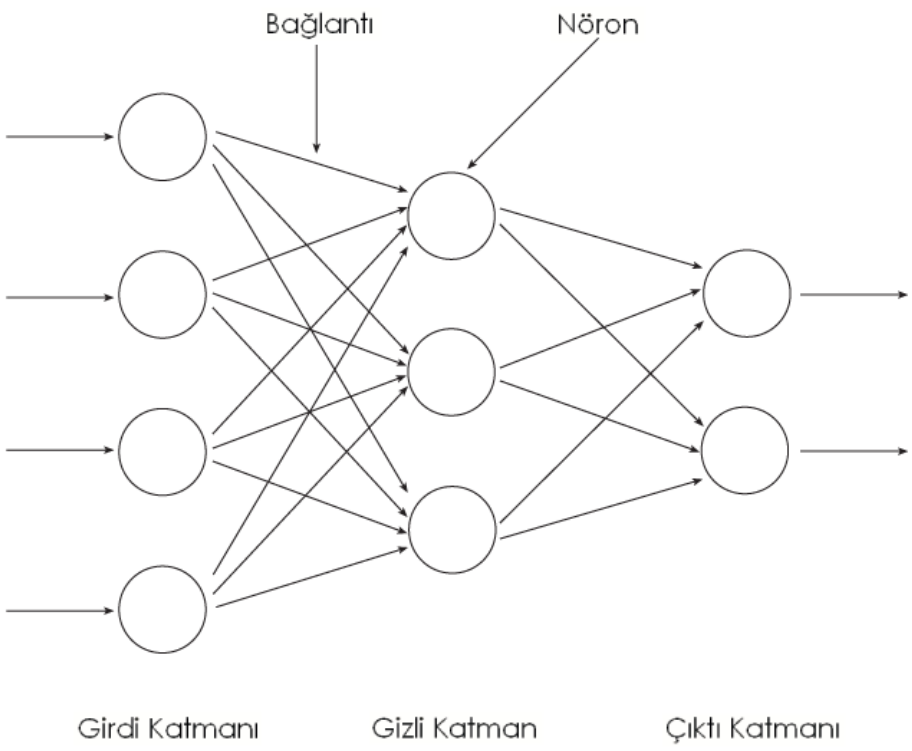
2.2. Yapay Sinir Hücreleri

Yapay sinir ağlarının temel elemanları, doğal nöronların işleyişlerini simüle eden **Yapay Nöronlardır**. Bu nöronlar, aralarındaki bağlantılar oluşturularak ve katmanlar halinde gruplandırılarak yapay sinir ağları oluşturulmaktadır. Şekil 2.2, McCullogh ve Pitts (1943) tarafından tanımlanan, biyolojik nöronun matematiksel gösterimidir.



Şekil 2.2 Yapay Sinir Hücresi

Yapay sinir hücreleri ayrıca **Düğüm** (Node), **Birim** (Unit) veya **İşlemci Eleman** (Processing Unit) olarak da adlandırılmaktadır. Nöronların aynı doğrultuda bir araya gelmeleriyle **Katmanlar** oluşur. Bir yapay sinir ağında üç tür katman bulunur. **Girdi Katmanı**, **Ara/Gizli Katman** ve **Çıktı Katmanı**. Katmanların belirli bir düzende dizilmesiyle **Ağlar** (Networks) oluşur. Ağlar, katmanların sayısı, dizilişleri ve öğrenme algoritmalarına göre isimler alırlar. Şekil 2.3'de üç katmanlı bir ağ örneklendirilmiştir.



Şekil 2.3 Yapay Sinir Ağı

Tek katman ya da tek eleman içeren bazı başarılı ağlar oluşturulabilmesine rağmen çoğu uygulamalar en az üç katman (girdi katmanı, gizli katman ve çıktı katmanı) içeren ağlara ihtiyaç duymaktadır. Girdi katmanı, dışarıdan girdileri alan nöronları içerir. Ayrıca, önemli olan bir nokta, girdi katmanındaki nöronların girdi değerleri üzerinde bir işlem uygulamamasıdır. Sadece girdi değerlerini bir sonraki katmana iletirler ve bu yüzden de bazı araştırmacılar tarafından ağların katman sayısına dahil edilmezler.

Çıktı katmanı ise çıktıları dış ortama veya başka bir ağa ileten nöronları içeren katmandır. Girdi ve çıktı katmanları tek katmandan oluşurken bu iki katman arasında birden fazla gizli katman bulunabilir. Bu gizli katmanlar çok sayıda nöron içerirler ve bu nöronlar tamamen ağ içindeki diğer katmanların nöronları ile bağlantılıdırlar. Çoğu ağ türünde, gizli katmandaki bir nöron sadece bir önceki katmanın tüm nöronlarından sinyal alır. Nöron işlemini yaptıktan sonra ise çıktısını bir sonraki katmanın tüm nöronlarına gönderir.

Bazı ağlarda, bir nöron aynı katmandaki başka nöronlara **Engel** (Inhibit) oluşturabilir. Bu işlem, **Yanal Engelleme** (Lateral Inhibition) veya **Rekabet**, **Yarışma** (Competition) olarak adlandırılır ve en çok çıktı katmanında kullanılırlar. Diğer bir bağlantı şekli ise **Geri Beslemedir** (Feedback). Geri besleme, bir katmanın çıktısının bir önceki katmana veya girdi katmanına gönderilmesidir.

Nöronların diğer nöronlara bağlanma şekli ağın çalışmasını önemli derecede etkilemektedir.

2.3. Yapay Sinir Hücrelerinin Bileşenleri

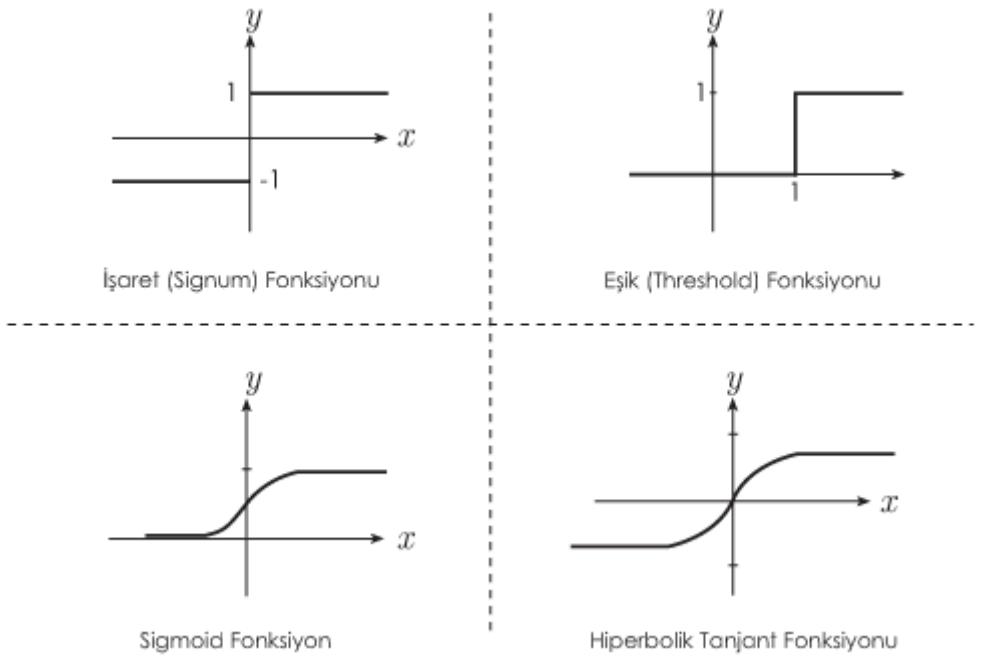
Dış ortamdan gelen bilgiler, girdi katmanına gelir. Bu bilgiler, ağın öğrenmesi istenen bilgilerdir. Bilgilerin ağa veriliş şekli eldeki veri setine göre değişiklik gösterir.

Girdi katmanından giren her bilgi (x_i) eş zamanlı olarak gizli katmana iletilir. Farklı katmanlardaki nöronlar arasındaki bağlantıların **Ağırlık Değerleri** (w_i) vardır. Ağırlık değerleri nöronlar arasındaki bağlantının gücünü belirtir.

Başlangıç ağırlık değerleri belirlendikten sonra, girdi katmanındaki hücrelerden gelen her bilgi (x_i) kendisine ait bağlantının ağırlık değeri (w_i) ile çarpılıp **Birleştirme (Toplama) Fonksiyonu** yardımıyla gizli katmandaki her bir hücrenin toplam **Net Girdi** (s) değeri bulunur. Net girdiyi oluşturan birleştirme fonksiyonu, kullanılan ağın yapısına göre ağırlık değerleri ile çarpılmış girdi değerlerinin toplanması, ortalaması, en büyüğü veya en küçüğünün alınması gibi birleştirme işlemlerinden herhangi biri olabilir. Bir problem için en uygun birleştirme fonksiyonu çeşidini bulmak için herhangi bir formül yoktur. Kullanılacak birleştirme fonksiyonu genellikle deneme yanılma yoluyla bulunmaktadır. Toplama işleminin yürütüldüğü basit bir birleştirme fonksiyonu Denklem (2.1)'deki gibi olabilir.

$$s = \sum x_i w_i \qquad \textbf{(2.1)}$$

Sonraki aşamada birleştirme fonksiyonunun çıktısı **Transfer Fonksiyonuna** gönderilir. Bu fonksiyon, aldığı değeri bir algoritma ile gerçek bir çıktıya dönüştürür. Kullanılan transfer fonksiyonuna göre çıktı değeri genellikle [-1,1] veya [0,1] arasındadır. Genellikle doğrusal olmayan bir fonksiyondur. Doğrusal olmayan transfer fonksiyonlarının kullanılması yapay sinir ağlarının karmaşık ve çok farklı problemlere uygulanmasını sağlamıştır. Sıklıkla kullanılan transfer fonksiyonları **Doğrusal** (Linear), **Adım/İşaret** (Step/Signum), **Eşik** (Threshold), **Sigmoid**, **Hiperbolik Tanjant, Lojistik** vb. fonksiyonlardır. En çok kullanılan transfer fonksiyonlarının bazıları Şekil 2.4'de grafik şeklinde gösterilmiştir.



Şekil 2.4 Sık Kullanılan Transfer Fonksiyonları

Doğrusal Fonksiyon:

$$F(s) = s \tag{2.2}$$

Adım/İşaret Fonksiyonu:

$$y = F(s) = \begin{cases} 1 & s > 0 \\ -1 & s \leq 0 \end{cases} \tag{2.3}$$

Eşik Fonksiyonu:

$$y = F(s) = \begin{cases} 1 & s > \theta \\ 0 & s \leq \theta \end{cases} \tag{2.4}$$

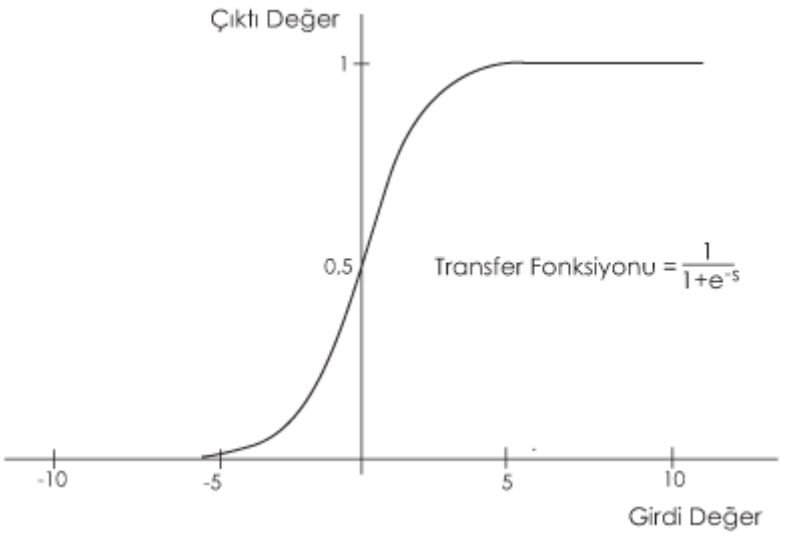
Sigmoid Fonksiyon:

$$y = F(s) = \frac{1}{1 + e^{-s}} \tag{2.5}$$

Hiperbolik Tanjant Fonksiyonu:

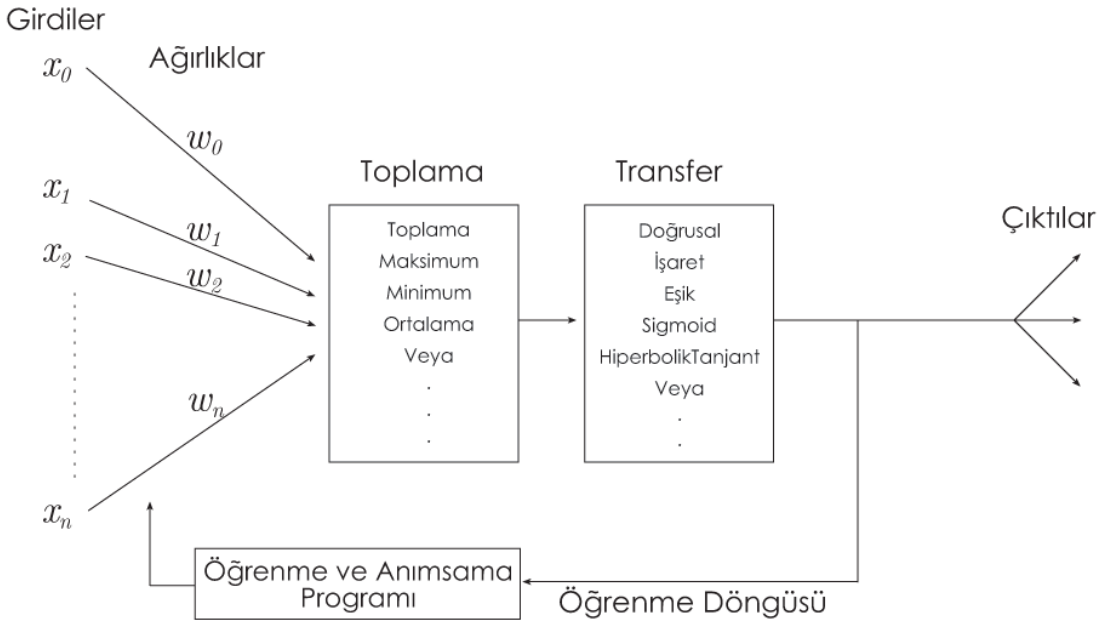
$$y = F(s) = \frac{e^{s} + e^{-s}}{e^{s} - e^{-s}} \tag{2.6}$$

Transfer fonksiyonunun sonucu genellikle işlem elemanının (nöronun) çıktısıdır. Transfer fonksiyonunun çalışma yapısı Şekil 2.5'de sigmoid fonksiyon kullanılarak örneklenmektedir.



Şekil 2.5 Sigmoid Transfer Fonksiyonu

Transfer fonksiyonu tarafından belirlenen çıktı değeri, bir başka nörona veya dış ortama gönderilir. Biyolojik sinir hücrelerinde olduğu gibi, her işlemci eleman birden çok girdi değerine sahipken, tek bir çıktı değerine sahiptir. Şekil2.6'da detaylı bir Yapay sinir hücresi verilmiştir.



Şekil 2.6 Detaylı Bir Yapay Sinir Hücresi

Transfer fonksiyonundan sonra, eğer gerekliyse, çıktı değerine **Ölçekleme ve Sınırlandırma** işlemi yapılır. Transfer değeri bir ölçeklendirme değeri ile çarpılarak çıktı değeri dengelenir. Bu işlem çıktı değerinin belirli bir değer aralığında olmasını garantiler.

Bazı ağ yapılarında, hangi nöronun aktif olacağı yani çıktı üreteceği veya hangi nöronun öğrenme sürecinde aktif rol oynayacağının belirlenmesi için nöronlar kendi aralarında yarışırlar. Genellikle çıktı değeri transfer fonksiyonunun çıktı değerine eşittir fakat bu yarışma sırasında transfer fonksiyonun çıktı değeri üzerinde bazı dönüşümler uygulanabilir. Bunun için **Çıktı Fonksiyonu** kullanılmaktadır.

Bir çok öğrenen ağda çıktı değeri ile istenen çıktı değeri arasında farklar oluşmaktadır. Bu hata değeri ağ yapısına uygun olarak **Performans Fonksiyonu** tarafından hesaplanırlar. Bir çok basit ağ yapısında bu hata değeri değiştirilmeden kalırken, bazı ağlarda negatif değerlerinden kurtulmak için hatanın karesi alınır.

Yapay nöronun hata değeri bir başka işlem elemanının **Öğrenme Fonksiyonuna** iletilir. Genellikle bu iletilme **Geri Yayılım Algoritması'**nda olduğu gibi geriye doğru yapılır. Geriye yayılan değer, hata değeri olabilirken, yine ağ yapısına göre transfer fonksiyonunun türevi ile çarpılmış hali gibi hata değerinin ölçeklenmiş değeri de olabilir. Normalde bu geri yayılmış değer öğrenme fonksiyonu tarafından ölçeklendirildikten sonra tüm girdi bağlantılarının değerleri ile çarpılır ve böylece bir sonraki öğrenme sürecinin yeni ağırlıkları oluşturulur.

Öğrenme fonksiyonunun amacı, belirli bir öğrenme algoritması yardımı ile, bağlantıların ağırlık değerlerini değiştirerek minimum hata ile istenen çıktı değerlerine ulaşmaktır.

2.4. Ağ Eğitimi

Öğrenme; Gözlem, eğitim ve hareketlerin doğal yapıda meydana getirdiği davranış değişikliği olarak tanımlanmaktadır (Saraç, 2004, s.29). Öğrenme tanımlarından bir diğeri Simon tarafından önerilen ve geniş kabul gören bir tanımdır. Bu tanıma göre öğrenme, "Zaman içinde yeni bilgilerin keşfedilmesi yoluyla davranışların iyileştirilmesi sürecidir" (Öztemel, 2003, s.21).

Makine öğrenmesi: "Bilgisayarların bir olay ile ilgili bilgileri ve tecrübeleri öğrenerek, gelecekte oluşacak benzeri olaylar hakkında kararlar verebilmesi ve çözümler üretebilmesidir" (Öztemel, 2003, s.21).

Belirli bir uygulamaya yönelik bir ağ yapılandırıldıktan sonra ağın eğitilme işlemine başlanır. Eldeki veriler ağa sunularak ağın öğrenmesi sağlanır. Problemin türüne ve kullanılan ağın yapısına göre önceden belirlenen öğrenme kuralı kullanılarak ağın bağlantı ağırlıkları değiştirilir. Amaç ağa gösterilen örnekler için doğru çıktıları üretecek ağırlık değerlerinin belirlenmesidir. Genel olarak, başlangıç ağırlıkları rastsal olarak seçilir.

Eğitim için kullanılan veri setine **Eğitim Seti** denir. Eğitim boyunca aynı eğitim seti defalarca ağa verilerek ağırlıkların en uygun seviyeye gelmesi sağlanır. **İterasyon** (Epoch) kelimesi, tüm eğitim setinin ağa gösterilmesi olarak ifade edilir. Eğitim süreleri genelde iterasyon ile ölçülür.

Ağın ağırlıklarının değiştirilmesinde iki tür strateji uygulanır.

1. Hata bilgileri toplanarak her bir iterasyon sonrasında ağırlıklar değiştirilir.
2. Ağırlıklar veri setindeki her bir veriden sonra değiştirilir.

Yapay sinir ağında ağırlıkların doğru değerlere ulaşması örneklerin temsil ettiği problem konusunda ağın genellemeler yapabilme yeteneğine kavuşması demektir. Genelleme, yapay sinir ağının eğitiminde kullanılmamış ancak aynı örnek uzayından gelen girdi-çıktı örneklerini doğru sınıflandırabilme yeteneğidir. Ağın bu genelleştirme özelliğine kavuşması işlemine ağın öğrenmesi denilmektedir.

Yapay sinir ağının eğitiminin tamamlanmasının ardından öğrenip öğrenmediğini (performansını) ölçmek için yapılan denemelere ise ağın test edilmesi denmektedir. Bazı paket programlar ağ eğitilirken bir yandan da **Doğrulama Seti** (Validation Set) ile ağın her iterasyonda ne kadar öğrendiğini test ederler ve öğrenim kriteri olarak doğrulama setinin hata değerlerini kullanırlar.

Eğitim setindeki verilerle istenilen başarıya ulaşılmışsa, ağ, daha önce görmediği verilerle test edilir. Test işleminin yapıldığı veri setine **Test Seti** denir. Test işleminde ağın ağırlık değerleri değiştirilmemektedir. Örnekler ağa gösterilmekte ve ağ, eğitim sonucunda belirlenen ağırlık değerlerini kullanarak daha önce görmediği bu örnekler için çıktılar üretmektedir. Elde edilen çıktıların doğruluk dereceleri ağın öğrenmesi hakkında bilgi vermektedir. Sonuç ne kadar iyi olursa eğitimin performansı da o kadar iyi demektir. Eğer test seti ile istenilen sonuçlara ulaşılırsa ağın eğitimi biter.

Bazı ağlarda ise eğitim gerçek verilerle sürekli olarak devam eder. Bu noktada ağlar öğrenme türlerine göre ikiye ayrılabilirler.

Çevrimdışı Öğrenme: Çevrimdışı (Offline) öğrenme kuralına dayalı ağlar eğitildikten sonra gerçek hayatta kullanıma alındığında artık öğrenme olmamaktadır. **Delta Öğrenme Kuralı** bu tür öğrenmeye örnek olarak verilebilir.

Çevrimiçi Öğrenme: Çevrimiçi (Online) öğrenme kuralına göre öğrenen ağlar, gerçek zamanda çalışırken bir taraftan fonksiyonlarını yerine getirmekte, bir taraftan da öğrenmeye devam etmektedirler. **ART** ve **Kohonen's SOM** ağlarında kullanılan **Kohonen Öğrenme Kuralı** bu öğrenme kuralına örnek olarak verilebilir.

2.4.1. Öğrenme Metotları

Genel olarak **Danışmanlı** (Supervised) ve **Danışmansız** (Unsupervised) ve **Takviyeli** (Reinforcement) olmak üzere üç tür öğrenme metodu vardır. Danışmanlı Öğrenme, ağın istenilen çıktı değerlerine ulaşılabilmesi için dışardan müdahale veya bir iç mekanizmaya ihtiyaç duyar. Danışmansız Öğrenme ise dış müdahale olmaksızın, girdilerin ağ tarafından analiz edilmesi ve bu analiz sonucunda bağlantıların ağırlık değerlerinin değiştirilmesidir. Takviyeli öğrenme her iki stratejiyle de benzerlikler taşır.

2.4.1.1. Danışmanlı Öğrenme

Danışmanlı öğrenmede, girdi ve çıktı değerlerinin her ikisi de ağa gösterilir. Girdi değerleri ağ tarafından işlenerek istenilen çıktı değerleri ile ağın çıktı değerlerini karşılaştırır. Aradaki fark **Hata** olarak ele alınır. Performans fonksiyonu ile hesaplanan bu hata değerini minimize etmek için hesaplanan değer sisteme geri verilir. Ağ, kendi çıktı değerini istenilen çıktı değerine yaklaştırmak için hücre bağlantılarının ağırlıklarını değiştirir. Bu sayede girdilerle çıktılar arasındaki ilişkiler öğrenilmektedir.

Öğrenme süreci, hata değerleri istatistiksel olarak kabul edilebilir seviyeye ininceye kadar devam eder. **Çok Katmanlı Algılayıcı** (Multilayer Perceptron) ağları bu tür öğrenme metodunu kullanan ağlara örnektir.

2.4.1.2. Danışmansız Öğrenme

Danışmansız Öğrenme Metodunda, ağa sadece girdiler verilir, istenilen çıktı değerleri verilmez. Örneklerdeki parametreler arasındaki ilişkiyi kendi kendisine öğrenmesi beklenir. Ağ, girdi değerlerini gruplandırmak için yapacağı değişikliklere kendi karar verir. Daha çok sınıflandırma problemlerinin çözümünde kullanılır. Öğrenme bittikten sonra çıktıların ne anlama geldiğini belirtmek için bir kullanıcıya ihtiyaç vardır. Bu yöntem **Kendi Kendine Öğrenme** (Self-Organization) veya **Adaptasyon** olarak bilinir.

Günümüzde danışmansız öğrenmenin çalışma sistemi tam olarak anlaşılamamış durumdadır ve bu konudaki araştırmalar hala devam etmektedir. Bu tür öğrenmenin öncü araştırmacılarından birisi Tuevo Kohonen'dir. Kohonen, doğru cevabı bilmenin faydalarından yararlanmadan öğrenen bir ağ geliştirmiştir. **Kohonen's SOM** (Self Organizing Map) adıyla bilinen bu ağ, bir çok sayıda bağlantı ve tek katmana sahip olması nedeniyle biraz sıra dışı sayılabilir. **ART** ağları danışmansız öğrenme kuralını kullanan en önemli ağlardan biridir.

2.4.1.3. Takviyeli Öğrenme

Takviyeli öğrenmede, danışmanlı öğrenmede olduğu gibi bir eğitici yardımcı olur. Farklı olarak sisteme sadece girdi değerleri verilir, istenilen çıktı değerleri gösterilmez. Ağın kendi çıktılarını oluşturması beklenir ve eğitici tarafından çıktı değerinin ne derece doğru olduğunu belirten bir sinyal, skor veya derece bildirilir. Ağ, eğiticiden aldığı sinyalle bağlantılarının ağırlık değerlerini değiştirerek öğrenme sürecini devam ettirir. **LVQ** ağları bu tür öğrenme metodu kullanan en önemli örnektir.

Optimizasyon problemlerini çözmek için Hinton ve Sejnowski'nin geliştirdiği **Boltzmann Kuralı** veya eğitimde **Genetik Algoritma** kullanımı takviyeli öğrenmeye örnek olarak verilebilirler.

2.4.1.4. Karma Öğrenme Metotları

Hem danışmanlı öğrenme hem danışmansız öğrenmeyi birlikte kullanan yapay sinir ağları da bulunmaktadır. Bu ağlarda ağırlıkların bir kısmı danışmanlı öğrenmeyle bir kısmı da danışmansız öğrenmeyle ayarlanmaktadır. **Radyal Tabanlı Yapay Sinir Ağları** (RBF) ve **Olasılık Tabanlı Yapay Sinir Ağları** (PBNN) bunlara örnek olarak verilebilir.

2.4.2. Performans Fonksiyonu

Öğrenme fonksiyonunun gerekli ayarlamaları yapabilmesi için yanılma payının biliniyor olması gerekmektedir. **Performans Fonksiyonu**, bu amaca yönelik olarak, o anki çıktı ile istenilen çıktı arasındaki farkı, hatayı, hesaplar ve gerekiyorsa bir **Dönüşüm** (Transformasyon) uygular. Bu hata, literatürde **Cari Hata** (Current Error) olarak adlandırılır ve bu hata veya dönüşümü sağlanmış hali (geri yayılma değeri) genellikle bir önceki katmana geri yayılır. Bu geri yayılma değeri, eğer gerekli ise, bir sonraki öğrenme döngüsünde öğrenme fonksiyonu tarafından bağlantıları ayarlamak için kullanılır.

2.4.3. Öğrenme Katsayısı

Öğrenme Katsayısı veya **Oranı** (Learning Rate), öğrenme sürecinin hızı ve işlevi açısından önemlidir. Yapay sinir ağlarının öğrenme gücü ile hızı ters orantılıdır. Basit bir şekilde, bir adım daha fazla öğrenme, daha düşük bir hız ve dolayısıyla daha fazla zaman anlamına gelmektedir. Diğer bir ifadeyle daha fazla hız; daha az öğrenme anlamına gelmektedir. Sonuç olarak, bir ağın ne kadar eğitileceği sorusu öğrenme oranına bağlıdır.

Öğrenme oranının belirlenmesinde ise istenen çözüm hızı, ağın karmaşıklık düzeyi, büyüklüğü, mimarisi, kullandığı öğrenme kuralı ve istenilen doğruluk derecesi gibi bir çok faktör rol oynar. Çoğu öğrenme fonksiyonu, öğrenme katsayısı için belirli standartlara sahiptir. Öğrenme katsayısı genellikle (0,1) gibi bir aralık içinde belirlenir. Ağın ağırlıkları bu katsayı kullanılarak arttırılır. Öğrenme katsayısının küçük değer alması, öğrenme sürecinin küçük adımlar halinde olmasını yani yavaş bir öğrenme sürecini getirir. Diğer taraftan maksimum doğruluk derecesine yakınsamayı getirebilir (Yurtoğlu, 2005, s.27) (Anderson ve McNeill, 1992, s.28).

Öğrenme hızını arttırmak amacı ile öğrenme katsayısı büyütülebilir. Büyük öğrenme katsayıları kullanıldığında ağ mutlak minimuma ulaşamayabilir. Bu duruma salınım denir. Salınımı engellemenin bir yolu da **momentum** teriminin kullanılmasıdır. Yerel çözümlere takılan ağların bir sıçrama ile yerel çözümlerden kurtulmasını sağlamaktadır.

2.4.4. Öğrenme Kuralları

Yapay sinir ağlarının eğitim süreci, belli kurallar çerçevesinde olmaktadır. Bu kurallara öğrenme kuralları adı verilmektedir. Yapay sinir ağlarının eğitimi için bir çok öğrenme kuralı kullanılmaktadır. Öğrenme kuralları, kullanılan yapay sinir ağlarının amacı ve ağın topolojisi ile doğrudan ilişkilidir. Ağırlıkların değiştirilmesi bu kurallara göre yapılmaktadır.

Bir çok öğrenme kuralı en eski öğrenme kuralı olan Hebb öğrenme kuralının varyasyonudur. Bunun dışında, araştırmacılar sürekli olarak yeni öğrenme kuralları geliştirmekte ve insanın öğrenmesine benzeyen çeşitli öğrenme kuralları geliştirmektedirler. Fakat makinelerin öğrenme işlemleri insanın öğrenme işlemiyle karşılaştırıldığında çok sınırlıdır. Öğrenme gerçekte, var olan öğrenme kuralları ile basitçe ifade edilemeyecek kadar karmaşık bir yapıya sahiptir. Bazı önemli öğrenme kuralları şunlardır:

2.4.4.1. Hebb Kuralı

İlk ve en bilinen öğrenme kuralı olan Hebb kuralı, 1949 yılında Donald Hebb tarafından geliştirilmiştir. Tanımı ***"The Organization of Behavior"*** kitabında anlatılmıştır. Kurala göre, bir hücre diğer bir hücreden bilgi alırsa ve her iki hücre de aktif ise her iki hücre arasındaki bağlantı kuvvetlendirilmelidir (Anderson ve McNeill, 1992, s.33). Eğer A hücresi B hücresini uyarmaya yetecek kadar yakınsa ve B hücresinin aktifleştirilmesinde sürekli olarak yer alıyorsa, A hücresinin etkinliği arttırılacak şekilde bir veya her iki hücrede de değişiklikler yapılır (Hebb, 1949, s. 49).

2.4.4.2. Hopfield Kuralı

Hebb kuralına benzemektedir. Bağlantıların ne kadar kuvvetlendirilmesi veya zayıflatılması gerektiği belirlenir. Eğer beklenen çıktı ve girdilerin her ikisi de aktif ise bağlantı öğrenme katsayısı nispetinde kuvvetlendirilir. Pasif ise zayıflatılır. Ağırlıkların kuvvetlendirilmesi veya zayıflatılması sırasında kullanılan öğrenme katsayısı kullanıcı tarafından sabit olarak atanmaktadır.

2.4.4.3. Delta Kuralı

Bu kural, Hebb kuralının biraz daha geliştirilmiş şeklidir. Bu kurala göre beklenen çıktı ile gerçekleşen çıktı arasındaki farkları azaltmak için bağlantıların ağırlık değerlerinin sürekli olarak değiştirilmesi gerekmektedir. Ağın ürettiği çıktı ile üretilmesi gereken çıktı arasındaki farkların karelerinin ortalamasının minimize edilmesidir.

Bu kural, **Widrow-Hoff Öğrenme Kuralı** veya **En Küçük Kareler Öğrenme Kuralı** (Least Mean Square, LMS) diye de bilinir.

Delta Kuralı şu şekilde çalışır: Çıktı katmanındaki delta hatası transfer fonksiyonunun türevi ile dönüştürülerek (Transformation) bir önceki katmana iletilir. Bu iletim sonucunda bir önceki katmanın girdi bağlantılarının katsayıları değiştirilir. Yani hata her seferinde bir önceki katmana geri yayılır. Bu geri yayılım işlemi ilk katmana ulaşıncaya kadar devam eder. Bu öğrenme kuralını kullanan ağlara **Geri Yayılım Ağları** denir. Geri yayılım adını hata teriminin çıktı katmanından ilk katmana kadar geriye doğru iletilmesinden almıştır.

Delta kuralı kullanılırken dikkat edilmesi gereken en önemli unsur, girdi setindeki verilerin rasgele dağılmış olması gerekmektedir. Eğitim setinin düzgün sırada olması veya yapısal olarak düzgün olması, istenilen doğruluğa ulaşmaya engel teşkil etmekte ve ağın öğrenmesini zorlaştırmaktadır.

2.4.4.4. Dereceli Azaltma (Gradient Descent) Kuralı

Bu kuralda, delta kuralında olduğu gibi delta hatasının değiştirilmesi için transfer fonksiyonunun türevi kullanılır. Öğrenme oranı bir sabitle çarpılarak ağırlık değiştirilir. Durağan duruma çok yavaş gelse de çok kullanılan bir kuraldır. Araştırmalar, farklı katmanlarda farklı öğrenme oranları kullanıldığı taktirde öğrenme işleminin daha hızlı olduğunu, durağan duruma daha hızlı yaklaşıldığını göstermektedir.

2.4.4.5. Kohonen Kuralı

Teuvo Kohonen (1982) tarafından, biyolojik sistemlerdeki öğrenmeden esinlenilerek geliştirilen bu öğrenme kuralına, **Yarışmacı Öğrenme Kuralı** (Competitive) da denmektedir.

Bu kuralda işlemci elemanlar, ağırlıklarının ayarlanması için yarışmaktadırlar. Hebb kuralından farklı olarak bir seferde yalnız bir işlemci elemanın, yani yalnızca kazanan nöronun bağlantı ağırlıkları değiştirilmektedir. En uygun çıktıya sahip işlemci elemanın kazandığı bu kuralda, kazanan işlemci eleman, kendisine komşu diğer işlemci elemanların ağırlıklarının değiştirilmesine de izin vermektedir. Sadece kazanan elemanın çıktı üretmesine ve komşu hücreleri ile birlikte ağırlıklarının değiştirilmesine izin verilir.

Komşu sayısı eğitim süresince değişiklik gösterir. Eğitim süreci boyunca en geniş komşu tanımından en dar komşu tanımına inilir. Kazanan eleman girdi desenini en iyi ifade eden eleman olarak tanımlanır.

3. Bölüm: Yapay Sinir Ağlarında Sınıflandırma

Günümüze kadar geliştirilen YSA'ların sayıca çok olmaları, araştırmacıları YSA'ları çeşitli şekillerde sınıflandırmaya götürmüştür. Sınıflandırmalar sadece gösterme amaçlı olup başka anlamlar taşımamaktadır. Kimi araştırmacılar ağları öğrenme metotlarına göre, kimi yapılarına göre, kimi de kullanım amaçlarına göre sınıflandırmışlardır. Bu bölüm, YSA'ların sınıflandırılması hakkında bilgiler içermektedir.

3.1. Öğrenme Metotlarına Göre Sınıflandırma

Yapay sinir ağları, ağların eğitiminde kullanılan öğrenme metotlarına göre sınıflandırılırlar. Bölüm 2.4.1'de bahsedildiği gibi genel olarak Danışmanlı (Supervised), Danışmansız (Unsupervised) ve Takviyeli (Reinforcement) olmak üzere üç tür öğrenme metodu vardır.

- **Danışmanlı Öğrenme Metodu** kullanan ağlara örnek olarak Tek ve Çok Katmanlı Algılayıcılar (Single ve Multilayer Perceptron) gösterilebilir.
- **Danışmansız Öğrenme Metodunu** kullanan ağlara örnek olarak **SOM** (Self Organizing Map) ve **ART** (Adaptive Resonance Theory) ağları gösterilebilir.
- **Takviyeli Öğrenme Metodunu** kullanan ağlara örnek olarak **LVQ** (Learning Vector Quantization), **Boltzman Makinesı** (Boltzman Machine) ve eğitimde **Genetik Algoritma** kullanan ağlar gösterilebilir.
- **Radyal Tabanlı Fonksiyon** (RBF) ve **Olasılık Tabanlı Ağlar** (PBNN) **Karma Öğrenme Metodu** kullanan yapay sinir ağlarına örnektir.

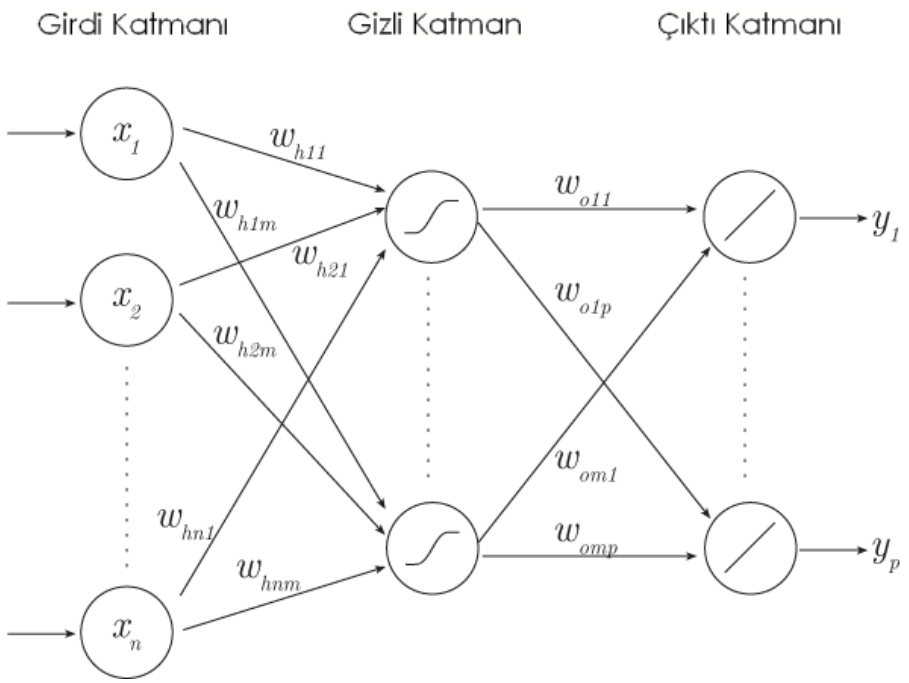
3.2. Yapılarına Göre Sınıflandırma

Yapay sinir ağları, yapılarına göre, **İleri Beslemeli** (Feedforward) ve **Geri Beslemeli** (Feedback) ağlar olmak üzere iki şekilde sınıflandırılırlar.

3.2.1. İleri Beslemeli Ağlar

İleri Beslemeli yapay sinir ağlarında nöronlar arasındaki iletişim giriş katmanından çıktı katmanına doğru tek yönlü bağlantılarla iletilir. Nöronlar bir katmandan diğer bir katmana bağlantı kurarlarken, aynı katman içerisindeki nöronlar birbiriyle bağlantılı değildir. Bu nedenle ileri beslemeli yapay sinir ağlarında, nöronlar arasındaki bağlantılar bir döngü oluşturmamakta ve bu ağlar girilen verilere hızlı bir şekilde çıktı üretebilmektedirler. İleri beslemeli ağlara örnek olarak **Çok Katmanlı Algılayıcılar** (Multi Layer Perceptron-MLP) ve **LVQ** (Learning Vector Quantization) ağları verilebilir.

İleri beslemeli yapay sinir ağlarında bir katmandaki tüm hücrelerin çıktıları bir sonraki katmandaki tüm hücrelere giriş olarak verilir. Giriş katmanı, dış ortamlardan aldığı bilgileri hiçbir değişikliğe uğratmadan gizli katmandaki hücrelere iletir. Bilgi gizli ve çıktı katmanında işlenerek ağın çıktısı belirlenir. Bu yapısı ile ileri beslemeli ağlar, doğrusal olmayan statik bir işlevi gerçekleştirir. İleri beslemeli üç katmanlı yapay sinir ağlarının, gizli katmanında yeterli sayıda hücre olmak kaydıyla, herhangi bir sürekli fonksiyonu istenilen doğrulukta yaklaştırabileceği gösterilmiştir (Saraç, 2004, s.27).



Şekil 3.1 İleri Beslemeli 3 Katmanlı YSA

Bu tip yapay sinir ağlarının eğitiminde genellikle **Geri Yayılım Öğrenme Algoritması** kullanılmaktadır. Şekil tanıma, sinyal işleme ve sınıflandırma gibi problemlerde genellikle bu topoloji uygulanmaktadır. Şekil 3.1'de ileri beslemeli üç katmanlı ağlara örnek verilmiştir.

3.2.2. Geri Beslemeli Ağlar

Geri Beslemeli veya **Yinelemeli** (Recursive) yapay sinir ağları, ileri beslemeli ağların aksine dinamik bir yapıya sahiptir. Geri beslemeli yapay sinir ağlarında, çıktı veya ara katmanlardaki nöronlar çıktılarını giriş veya önceki ara katmanlardaki nöronlara tekrar girdi olarak iletirler. Böylece bilgi hem ileri hem de geri yönde aktarılmış olmaktadır. Geri beslemeli yapay sinir ağları dinamik hafızaya sahiptir ve bir andaki çıktı hem o andaki hem de önceki girdileri yansıtmaktadır.

Geri beslemeli yapay sinir ağlarında, en az bir işlemci elemanın çıktısı, kendisine ya da diğer işlemci elemanlara girdi olarak verilmekte ve genellikle geri besleme bir **Geciktirme Elemanı** üzerinden yapılmaktadır. Geri besleme, bir katmandaki işlemci elemanlar arasında olduğu gibi katmanlar arasındaki işlemci elemanlar arasında da olabilmektedir. Bu yapısı sayesinde geri beslemeli yapay sinir ağları, doğrusal

olmayan dinamik bir davranış göstermektedir. Dolayısıyla, geri beslemenin yapılış şekline göre farklı yapıda ve davranışta geri beslemeli yapay sinir ağlı yapıları elde edilebilmektedir (Saraç, 2004, s.28).

Geri beslemeli yapay sinir ağları karmaşık bir çalışma düzeneğine sahiptir ancak dinamik hafızaları nedeniyle önceden tahmin uygulamalarında ve sınıflandırma işlemlerinde başarılı sonuçlar vermektedirler.

3.3. Kullanım Amaçlarına Göre Sınıflandırma

Araştırmacılar YSA'ları kullanım amaçlarına göre sınıflandırmışlardır. YSA'ların kullanım alanları Bölüm 1.4'de incelenmiştir. Çalışmanın bu kısmında kullanım alanları ve amaçlarına göre hangi topolojilerin kullanıldığı anlatılacaktır. Yapay sinir ağlarının kullanım amaçları 8 maddede toplanabilir. Tablo 1.3'de kullanım amaçlarına göre bazı YSA topolojileri listelenmiştir.

Tablo 3.1 Kullanım Amaçlarına Göre YSA Topolojileri (Anderson-McNeill, 1992, s.31)

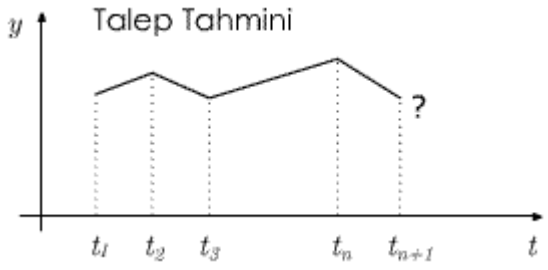
Tahmin-Öngörü	– Geri Yayılım – Yönlendirilmiş Rastsal Tarama (Directed Random Search) – Yüksek Dereceli Sinir Ağları – Geri Yayılım içinde SOM – Radyal Tabanlı Fonksiyon – Elman Ağı – Jordan Ağı
Fonksiyon Yaklaştırma	– Geri Yayılım – RBF
Desen (Örüntü) Sınıflandırma	– LVQ – ART – Olasılık Tabanlı Sinir Ağları – Tek veya Çok Katmanlı Algılayıcı – Boltzmann Makinesi – RBF
Veri İlişkilendirme	– Kohonen's SOM – Hopfield Ağları – Boltzmann Makinesi – Hamming Ağları – Çift Yönlü İlişkili Hafıza – Spatio-Temporal Desen Tanıma – ART
Kavramlaştırma / Kümeleme	– LVQ – SOM

	– ART
Veri Filtreleme	– Yeniden Dolaşım (Recirculation)
Optimizasyon	– Geri Yayılım – Olasılık Tabanlı Sinir Ağları – RBF
Kontrol	– Geri Yayılım – LVQ – RBF

Tablo 1.3, ağ kategorileri arasındaki farkları ve hangi topolojinin hangi amaçlarla kullanıldığını göstermektedir. Tablo tüm ağ topolojilerini değil, sadece çok kullanılan bazı topolojilerini göstermektedir. Bazı ağlar, birden çok problem tipinin çözümünde kullanılmaktadır. **İleri Beslemeli Geri Yayılım** (Feedforward Back-Propagation) ağları hemen hemen tüm problemlerin çözümünde kullanılmaktadır ve en çok kullanılan ağ topolojisidir (Jain, Mao ve Mohiuddin, 1996, s.8) (Anderson ve McNeill, 1992, s.31) .

3.3.1. Tahmin / Öngörü Yapma

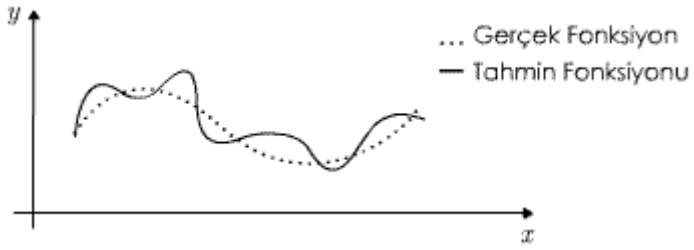
Girdi değerlerini, tahmin çıktıları üretmek için kullanan ağlardır. Günümüzde geliştirilen yapay sinir ağları en çok geleceğin öngörülmesinde, belirsizlik altında belirli bir konunun tahmin edilmesinde, kaynakların kullanımında veya belirli süreçlerin önceliklerinin belirlenmesinde, hava tahmini, ekonomik göstergelerin tahmini, hisse senedi tahmini, talep tahmini, hastanın kanser riski taşıyıp taşımadığı gibi tahmini bilgilerin üretilmesi için kullanılırlar. Tahmin amaçlı kullanılan ağların bazıları şunlardır: **Geri Yayılım** (Back-Propagation), **Yönlendirilmiş Rastsal Tarama** (Directed Random Search), **Yüksek Dereceli Sinir Ağları** (Higher Order Neural Networks), **Geriyayılım içinde SOM** (Self Organizing Map into Back-Propagation), **Radyal Tabanlı Fonksiyon** (Radial Basis Function, RBF)



Şekil 3.2 Tahmin İşleminin Şematik Gösterimi

3.3.2. Fonksiyon Yaklaştırma

Fonksiyon Yaklaştırma (Function Approximation) işlemi, eldeki bir çok veri çiftini işleyerek veri çiftlerinin oluşturduğu bilinmeyen fonksiyonu tahmin etmeye çalışmaktır. Bir çok mühendislik ve bilim alanında problemlerin modellenmesinde, karşılaşılan fonksiyonların tahminlerinde kullanılmaktadır. "Doğrusal olmayan modellerde kullanılan belirli fonksiyonel yapılar, veriyi üreten fonksiyonun genellikle yapay sinir ağlarının ima ettiğinden farklı olduğu ve bu yüzden yapay sinir ağlarının kullanılması için gerekli ekonometrik teorinin eksik tanımlı doğrusal olmayan modeller için olduğunu ima etmektedir. Bunların ön tanımlı yapıları dayanıksızken, yapay sinir ağları herhangi bir sürekli fonksiyona veya türevlerine yakınsama yeteneğine sahiptir ve bu yüzden **Evrensel Fonksiyon Yakınsayıcı Yöntem** (Universal Function Approximators) olarak tanımlanmaktadırlar." (Beltratti, 1996). **Geri Yayılım** ve **RBF Ağları** örnek olarak gösterilebilir.

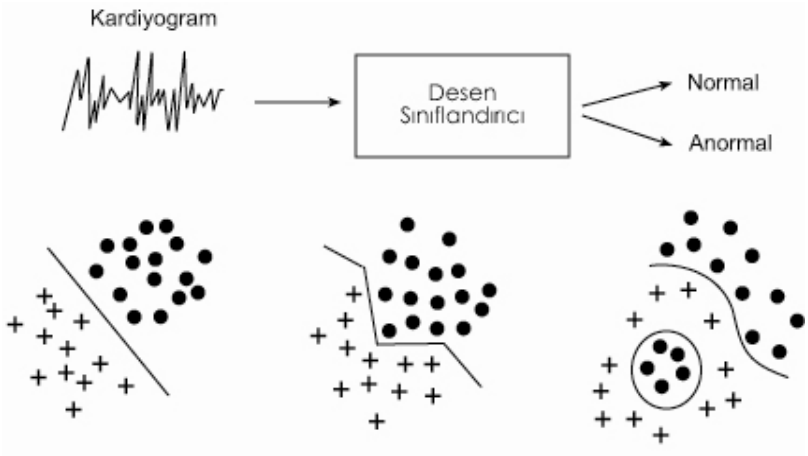


Şekil 3.3 Fonksiyon Yaklaştırma İşleminin Şematik Gösterimi

3.3.3. Desen Sınıflandırma

Yapay sinir ağlarının bir diğer kullanım alanı da **Desen/Örüntü Sınıflandırma** ve **Tanımadır** (Patern Classification and Recognition). Ses dalgaları, resim, kamera görüntüsü, el yazısı gibi desen içeren girdi değerlerini önceden belirlenmiş

desenlerle tanımlamaya çalışan ağlardır. Bir video görüntüsünün veya resmin neye/kime ait olduğunu, bir makinenin ürettiği ürünlerin hata grafiklerinin oluşturduğu desenleri inceleyerek makinenin hatalı çalışıp çalışmadığının kontrolü, laboratuar ve tıbbi görüntüleme cihazlarının ürettiği verilerden hastalık teşhisi, EEG ve ECG dalgalarını sınıflandırma, kan hücreleri sınıflandırma, yaklaşan nesnenin tespiti gibi konularda kullanılmaktadırlar. Bu amaçla kullanılan ağlar şunlardır: **LVQ** (Learning Vector Quantization), **ART** (Adaptive Resonance Theory), **Olasılık Tabanlı Sinir Ağları** (Probabilistic Based Neural Network, PBNN), **Geri Yayılım**, **Boltzmann Makinesi**, **SOM** ve **RBF Ağları**.

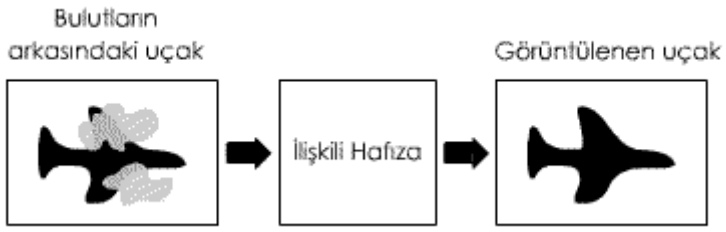


Şekil 3.4 Sınıflandırma İşleminin Şematik Gösterimi

3.3.4. Veri İlişkilendirme

Günümüzde kullanılan geleneksel bilgisayar sistemlerinde (Von-Neumann Modeli Hesaplama Yöntemi) bilginin içeriğinden çok saklandığı yerin adresi ile ilgilenilir. Eğer verinin bir kısmında bozukluk varsa veya adresin hesaplanmasında bir hata oluşursa doğru bilgiye ulaşmak imkansızlaşır. **İlişkili Hafızada** (Associative Memory) veya **İçerik Adresli Hafızada** (Content-Addressable Memory) veriye fiziksel adresi ile değil içerik adresi ile ulaşılır. Hafızadaki içeriğe, girdi bilgisi eksik de olsa, hafızadaki bilgi kısmen bozuk da olsa ulaşılarak çağrılması sağlanır. Görsel içeriğe ve ses içeriğine sahip çoklu ortam (Multimedya) veritabanlarında ve veri madenciliğinin metin tabanlı uygulamalardan kurtulmasını sağlayan çoklu ortam madenciliğinde (Multimedia Mining) oldukça sık kullanılmaktadır. Bulutların arkasındaki uçan bir nesnenin tanımlanmasında, farklı açılardan fotoğraflanmış bir suçlunun teşhisinde, eksik

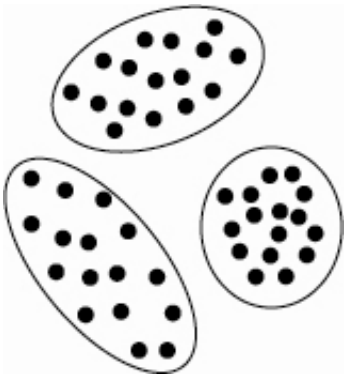
parçaları bulunun desenlerin tamamlanmasında kullanılmaktadır. **Veri İlişkilendirme** (Data Association) amacıyla kullanılan ağların bazıları şunlardır: **SOM**, **Hopfield Ağları**, **Boltzmann Makinesi**, **Hamming Ağları**, **Çift Yönlü İlişkili Hafıza** (Bidirectional Associative Memory), **Spatio-temporal Desen Tanıma** ve **ART Ağları**.



Şekil 3.5 Veri İlişkilendirme İşleminin Şematik Gösterimi

3.3.5. Veri Kavramlaştırma / Kümeleme

Veri Kavramlaştırma (Data Conceptualization), **Kümeleme** (Clustering) ve **Kategorize Etme** (Categorization) işlemine YSA terminolojisinde **Danışmansız Desen, Örüntü Tanıma** da denir. Eğitim için etiketlenmiş veya sınıflandırılmış veri setine ihtiyaç duymazlar. Veri setindeki benzer desenleri inceleyerek benzerliklerine göre veri setini olabilecek en iyi şekilde kümelemeye çalışırlar. Reklam amaçlı gönderilecek e-postaların veritabanındaki belirli özellikler taşıyan üyelere ulaştırılması, reklamın gösterileceği hedef kitlesinin oluşturulması gibi amaçlarla kullanılırlar. **Veri Madenciliği** ve **Veri Sıkıştırma** konularında da sıkça kullanılırlar. **LVQ**, **Kohonen's SOM** ve **ART Ağları** bu amaçla kullanılan ağlardandır.



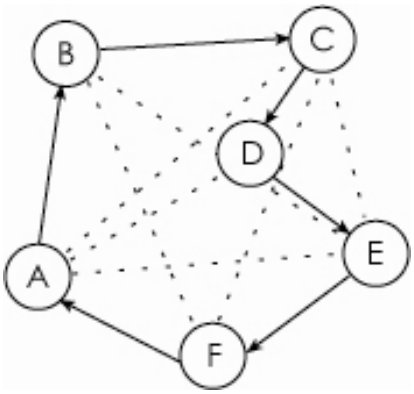
Şekil 3.6 Kümeleme İşleminin Şematik Gösterimi

3.3.6. Veri Filtreleme

Yapay sinir ağlarının kullanıldığı en önemli noktalardan biri **Veri Filtreleme**'dir (Data Filtering). Girdi sinyallerinin düzgünleştirilmesinde kullanılmaktadır. İlk ağlardan biri olan **MADALINE** bu kategoriye girer. Telefon hatlarındaki parazitlerin ve yankılanmanın giderilmesi amacıyla dinamik yankı önleyici devrelerde kullanılmıştır. Halen modemlerde **Dinamik Eşitleme** (Dynamic Equalization) tekniği ile kullanılmaktadır. Bu amaçla **Yeniden Dolaşım** (Recirculation) **Ağları** kullanılmaktadır.

3.3.7. Optimizasyon

Matematik, istatistik, mühendislik, bilgisayar, tıp ve ekonomi gibi bir çok bilim dalında optimizasyon problemleri ile karşılaşılır. Belirli kısıtlar altında, bir fonksiyonun maksimum veya minimum noktalarının bulunması veya optimum çözümlerinin üretilmesi optimizasyon problemlerinin amacıdır. **Gezgin Satıcı** (Travelling Salesman) problemleri, polinom zamanlı çözümü olmayan (NP-Hard, NP-Complete) problemler bu tür problemlere örnektir. **Geri Yayılım**, **Olasılık Tabanlı Sinir Ağları** (PBNN) ve **RBF Ağları** bu amaçla kullanılan ağlardan bazılarıdır.

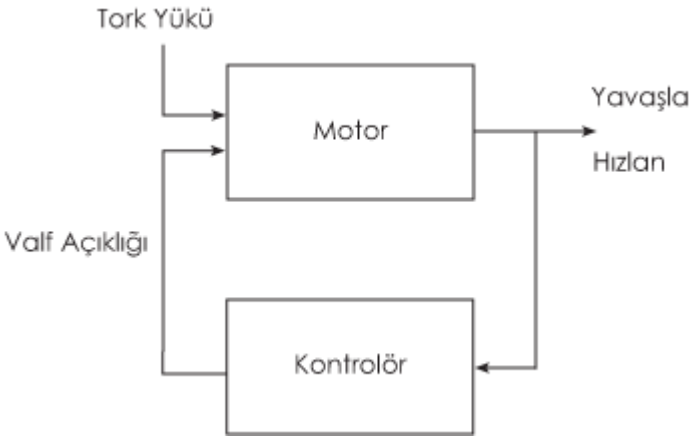


Şekil 3.7 Gezgin Satıcı Probleminin Şematik Gösterimi

3.3.8. Kontrol

Dinamik kontrol gerektiren uygulamalar, kontrol girdisi ile çıktısı arasındaki ilişkiyi eş zamanlı olarak inceleyip gerekli tepkileri anında verecek şekilde tasarlanırlar. Referans modeline uygun olarak istenilen çıktıların üretilmesi için kontrol girdilerinin

değiştirilmesi prensibine göre çalışırlar. Otomobillerdeki hız kontrolü ve virajlarda savrulmayı önleyen sistemler, bir makinenin ürettiği ürünlerin kalitelerini anlık olarak gözlemleyip makine ayarlarını yapan sistemler ve robotik sistemler buna örnek olarak gösterilebilir. Genellikle uzman sistemler ile birlikte kullanılırlar. Evlerimizdeki elektronik ve beyaz eşyalarda da halihazırda kullanılmaktadırlar. **Geri Yayılım, LVQ** ve **RBF Ağları** bu amaçla kullanılan ağlardandır.



Şekil 3.8 Kontrol İşleminin Şematik Gösterimi

4. Bölüm: Ağ Topolojileri

Nöronların bağlantı şekilleri, oluşturdukları katman sayısı ve katmanlar arasındaki veri iletim şekli ağın yapısını yani topolojisini belirler. Bir başka deyişle ağ oluşturmak için bir araya gelen nöronların bağlantı şekline **Ağın Topolojisi** (Fiesler, 1996, s.2) denir. Bir ağın topolojisi, ağın fonksiyonelliği ve performansı üzerinde önemli bir rol oynar. Yapı veya mimari de denmektedir. Ağın topolojisi ile, bir ağın nöronlarının dizilişi ve bağlantılarının yapısı ifade edilir.

Topoloji seçimi, çözülmek istenen problemle ilgilidir. Hangi ağ modelinin hangi problemin çözümü için daha uygun olduğunun bilinmesi oldukça önemlidir. Bu konu beşinci bölümde daha detaylı incelenecektir. Yapay sinir ağları ağda kullanılan topolojiye göre isimlendirilirler. Günümüzde çok fazla YSA topolojisinin kullanıldığı bilinmekte ama bu topolojilerin sayısı bilinmemektedir. Topolojiler, ağda kullanılan katman sayısı (tek katmanlı, çok katmanlı), öğrenme algoritması (danışmanlı, danışmansız, takviyeli), öğrenme kuralı, iletişim yönü (ileri beslemeli, geri beslemeli, yarışmacı) gibi belirleyici özellikleri ile isimlenirler. Tablo 4.1'de çok bilinen bazı YSA Topolojilerinin sınıflandırılması yapılmıştır.

Tablo 4.1: YSA Türlerinin Sınıflandırılması (Sarle, 1997)

A. Danışmanlı
   1. İleri Beslemeli
      i. Doğrusal
         - Hebb
         - Algılayıcı
         - Adaptif Doğrusal Eleman
         - Yüksek Dereceli Ağlar
         - Fonksiyonel Bağlantı (Functional Link)
      ii. Çok Katmanlı Algılayıcı
         - Geri Yayılım
         - Kademeli (Cascade) Korelasyon
         - Hızlı Yayılım
         - Esnek Yayılım (Resilient Propagation, RPROP)
      iii. RBF Ağları
         - Dikey En Küçük Kareler (Orthogonal Least Squares)
      iv. CMAC: Beyincik (Cerebellar) Model Artikülasyon Kontrolörü
      v. Sadece Sınıflandırma

- Doğrusal Vektör Niceleme (LVQ)
- Olasılık Tabanlı (PBNN)

vi. Sadece Regresyon
- Genel Regresyon Yapay Sinir Ağı (GNN)

2. Geri Beslemeli
   i. Çift Yönlü İlişkili Hafıza (BAM)
   ii. Boltzman Makinesi
   iii. Yinelemeli Zaman Serileri
   - Zaman Boyunca Geri Yayılım
   - Elman
   - Sonlu Dürtü Yanıtı (Finite Impulse Response, FIR)
   - Gerçek Zamanlı Yinelemeli Ağ
   - Yinelemeli Geri Yayılım
   - Zaman Gecikmeli Sinir Ağı (TDNN)
3. Yarışmacı
   i. ARTMAP
   ii. Fuzzy ARTMAP
   iii. Gaussian ARTMAP
   iv. Karşı Yayılım
   v. Neocognitron

B. Danışmansız
1. Yarışmacı
   i. Vektör Niceleme
   - Grossberg
   - Kohonen
   - Vicdani (Conscience)

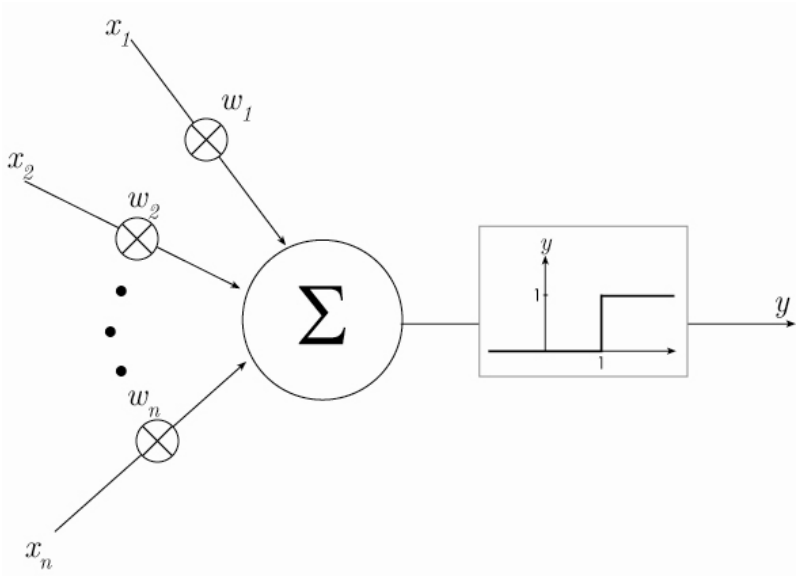
   ii. Self-Organizing Map
   - Kohonen
   - Yerel Doğrusal (Local Linear)

   iii. Adaptive resonance theory
   - ART 1
   - ART 2, ART 2-A
   - ART 3
   - Fuzzy ART

   iv. Diferansiyel Yarışmacı Öğrenme (DCL)
2. Boyut Düşürme
   i. Hebb
   ii. Oja
   iii. Sanger
   iv. Diferansiyel Hebb
3. Kendinden İlişkili
   i. Doğrusal Kendinden İlişkili
   ii. Kutudaki Beyin Durumu (Brain State in a Box)
   iii. Hopfield

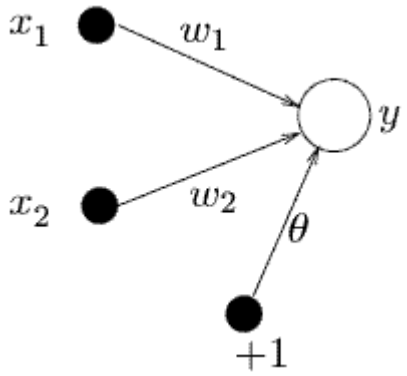
4.1. Tek Katmanlı Ağlar

Tek katmanlı, ileri beslemeli ağlar, bir veya birden fazla nörona sahip bir girdi ve bir çıktı katmanından oluşurlar. Girdi katmanındaki her nöron, çıktı katmanındaki tüm nöronlarla bağlantılıdır. Nöronların girdi değerleri bağlantıların ağırlıkları ile çarpılıp toplanır. Eğer bu toplam değeri belirli bir eşik değerini aşarsa nöron bir çıktı değeri üretir. Bu en basit yapay nörona **Eşik Mantık Birimi** (EMB) (Threshold Logic Unit-TLU) denir. McCulloch ve Pitts (McCulloch and Pitts, 1943) tarafından önerilen en basit EMB yapısı Şekil 4.1 'de gösterilmiştir.



Şekil 4.1 McCulloch and Pitts tarafından önerilen en basit EMB Modeli

n adet girdi *(x_1, x_2,....,x_n)* ve bağlantıların ağırlıkları *(w_1,w_2,....,w_n)* ile ifade edilir. En basit haliyle, tek işlemci elemanlı bir yapay sinir ağı Şekil 4.2'de gösterilmiştir.



Şekil 4.2 Tek işlemci Elemana Sahip Yapay Sinir Ağı Modeli (Kröse ve Smagt, 1996, s.23)

Çıktı katmanındaki nöronun girdi değeri, girdi katmanındaki nöronların ağırlıklı toplamı ile **Yanlı** (Bias) teriminin toplamından oluşur ve şu fonksiyonla ifade edilir:

$$y = F\left(\sum_{i=1}^{n} w_i x_i + \theta\right) \quad \textbf{(4.1)}$$

Ağın çıktısı, çıktı katmanındaki nöronun aktivasyon fonksiyonunun çıktısıdır *(y)*. x_i değerleri girdi katmanındaki nöronların değerlerini, w_i değerleri bağlantıların ağırlıklarını, *n* değeri girdi katmanındaki toplam nöron sayısını, θ değeri yanlı terimini (eşik değeri), *F* fonksiyonu ise aktivasyon fonksiyonunu ifade eder. Aktivasyon fonksiyonu Bölüm 2.3'te anlatıldığı gibi doğrusal, sigmoid, eşik, adım (işaret) fonksiyonu gibi bir fonksiyon olabilir. Eğer eşik fonksiyonu seçilmişse ağın çıktısı şu fonksiyona göre Denklem (4.2)'deki gibi, eğer adım fonksiyonu seçilmişse Denklem (4.3)'teki gibi hesaplanmaktadır:

$$y = F(s) = \begin{cases} 1 & s > \theta \\ 0 & s \leq \theta \end{cases} \quad \textbf{(4.2)}$$

$$y = F(s) = \begin{cases} 1 & s > 0 \\ -1 & s \leq 0 \end{cases} \quad \textbf{(4.3)}$$

Tek katmanlı ağın (Perceptron) eşik fonksiyonu ile kullanımı **Doğrusal Ayrım Fonksiyonunu** (Linear Discriminant Function) ifade eder. Şekil 4.2'deki örnekte doğrusal ayrım fonksiyonunu kullanan EMB, girdi desenlerini iki grup şeklinde sınıflandırmaktadır, yani girdi uzayını iki parçaya bölmektedir. Bu durumda ağ, sınıflandırma görevi görecektir, yani ağ, girdi desenlerinin belirlenen iki sınıftan birine ait olduğuna karar verecektir. Ağ, eğer toplam girdiler pozitif ise girdi deseninin çıktısını +1 yapacak ve deseni +1 sınıfına atayacak, eğer toplam girdiler negatif ise girdi desenini aynı şekilde -1 olarak atayacaktır. İki sınıf arasındaki ayrım, bu durumda şu denklemle ifade edilir:

$$w_1 x_1 + w_2 x_2 + \theta = 0 \quad \textbf{(4.4)}$$

EMB'nin çalışma şekli geometrik olarak gösterilebilir. Her çıktı deseni x_1 ve x_2 şeklinde iki bileşene sahiptir. Girdi değerlerinin oluşturduğu uzaya desen uzayı (Pattern Hyberspace) denir. Her desen uzayda kendi bileşenini oluşturarak bir nokta belirtir. *n* girdi olması durumunda uzay *n* boyutlu olacaktır. Açıktır ki, üçten fazla girdi olduğu takdirde desen uzayının çizimi imkansız hale gelecektir.

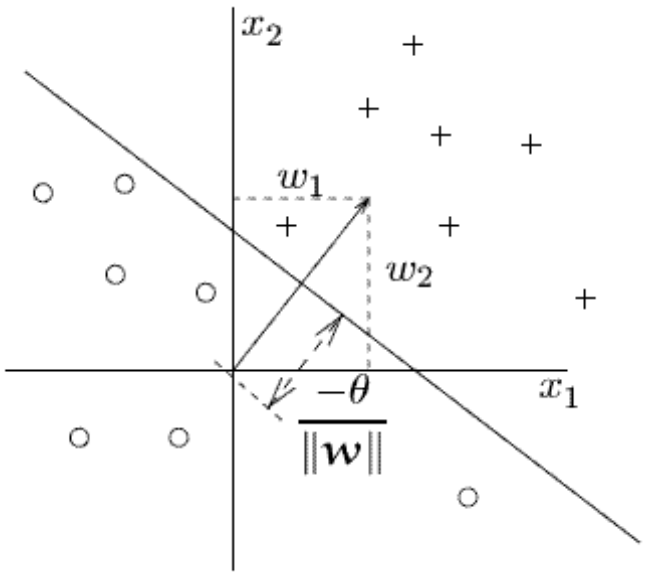
Denklem (4.5), şu şekilde tekrar yazılır:

$$x_2 = -\frac{w_1}{w_2} x_1 - \frac{\theta}{w_2} \qquad \textbf{(4.5)}$$

Bu denklem iki boyuttaki doğru denklemidir:

$$x_2 = ax_1 + b \qquad \textbf{(4.6)}$$

Bu doğruya **Karar Doğrusu** (Decision Line) denir. Denklem (4.5)'ten ağırlıkların oranının doğrunun eğimini, yanlı teriminin de doğrunun koordinat ekseninin başlangıç noktasından ne kadar uzaklaştığını belirlediği görülmektedir. **Doğrusal Eşik Sinir Ağı**'nın **Düzlem** (Hyperplane) üzerindeki geometrik şekli Şekil 4.3'te gösterilmiştir. Dikkat edilirse ağırlıklar girdi uzayında vektör olarak ifade edilirler ve ağırlık vektörü ayrım fonksiyonuna her zaman diktir. İkiden fazla boyutlarda **Karar Düzlemi** (Decision Hyperplane) haline gelir.



Şekil 4.3 Doğrusal Ayrım Fonksiyonunun Geometrik Gösterimi (Kröse ve Smagt, 1996, s.30)

Bu ağlarda öğrenmeden kastedilen, sınıfları birbirinden ayıran en uygun doğrunun bulunmasıdır. Tek katmanlı ağların eğitimlerinde iki tür yöntem kullanılmaktadır. **Algılayıcı Öğrenme Kuralı** ve **Delta** (veya **LMS**) **Kuralı**. Her iki yöntem de ağların ağırlıklarını değiştiren iteratif bir yöntemdir. Ağırlıkların değiştirilmesi doğrunun eğiminin değiştirilmesi demektir. *t* iterasyonundaki ağırlık değerleri Δw kadar değiştirilir ise;

$$w_i(t+1) = w_i(t) + \Delta w_i(t) \qquad \textbf{(4.7)}$$

olmaktadır. Öğrenme sırasında bu değişim her yinelemede gerçekleştirilerek doğrunun en uygun eğimi bulunmaya çalışılmaktadır. Ancak bu işlem yeterli olmayabilmektedir. Bu nedenle eşik değerinin de (θ, bias veya ayrım fonksiyonunun *y* eksenini kestiği nokta) değiştirilmesi gerekmektedir. Bu işlem, doğrunun sınıflar arasında kaymasına yardımcı olmaktadır. Böylece doğrusal ayrım fonksiyonunun konumu belirlenmektedir. Öğrenme sırasında ağırlıklarda olduğu gibi eşik değeri de her yinelemede $\Delta\theta$ kadar değiştirilmektedir. Bu durumda *t* iterasyondaki eşik değeri

$$\theta(t+1) = \theta(t) + \Delta\theta(t) \qquad \textbf{(4.8)}$$

denklemi ile değiştirilmektedir.

Tek katmanlı yapay sinir ağlarında önemli iki modelden bahsedilebilir. Bu modeller **Algılayıcı** (Perceptron) ve **Adaptif Doğrusal Eleman**'dır (Adaline/Madaline).

4.1.1. Tek Katmanlı Algılayıcı

Tek Katmanlı Algılayıcı (Single Layer Perceptron), F. Rosenblatt tarafından 1950'li yılların sonlarında ortaya atılmış bir modeldir. İlk YSA modellerinden olan algılayıcılar, son derece sınırlı olmalarına karşın daha sonra geliştirilen modellere temel olmaları ve mevcut algoritmalarla yakın ilişki içinde olmaları nedeniyle oldukça önemlidirler.

Algılayıcı en basit şekilde bir yapay sinir hücresinin birden fazla girdiyi alarak tek bir çıktı üretmesi temeline dayanmaktadır. Ağın çıktısı 1 veya 0 olmakta ve çıktı

değerinin hesaplanması adım fonksiyonu kullanılarak yapılmaktadır. Problemin türüne göre ağın çıktısı {-1 ,1}, kullanılan aktivasyon fonksiyonu da Bölüm 2.3'te anlatılan doğrusal, eşik, adım, tanjant hiperbolik, sigmoid ve lojistik fonksiyonu gibi fonksiyonlardan biri olabilir.

Algılayıcının **Lojistik Aktivasyon Fonksiyonu** (Logistic Activation Function) ile kullanımı **Lojistik Regresyon Modeli**'ni (Logistic Regression Model) oluşturur (Hosmer ve Lemeshow, 1989) (Sarle, 1994, s.3).

Algılayıcının **Eşik Fonksiyonu** ile kullanımı **Doğrusal Ayrım Fonksiyonu**'nu (Linear Discriminant Function) ifade eder (Hand, 1981) (Mclachlan, 1992), (Weiss ve Kulikowski, 1991) (Sarle, 1994, s.3).

Bu ağlarda ara katman bulunmaz. Algılayıcı, eğitilebilen tek bir işlemci elemandan oluşmaktadır. Burada eğitilebilirlikten kasıt w_i ağırlık değerlerinin değiştirilebilir olması demektir. İşlemci elemana girdi değerleri ve bu değerlere karşılık gelen çıktı değerleri gösterilerek öğrenme kuralına göre ağın çıktı değeri hesaplanmaktadır. Olması gereken çıktı değerine ulaşılana kadar ağırlıklar değiştirilmektedir. Bu noktada çıktı değeri ile istenilen değer arasındaki fark **Hata** (Residual) olarak belirtilir. Amaç toplam kullanılan performans fonksiyonunun çıktısının sıfır değerine indirilmesidir. Algılayıcıda, anlaşılacağı gibi ileri beslemeli ve danışmanlı öğrenme uygulanmaktadır.

Algılayıcı Öğrenme Kuralı: x eğitim seti örneklerinin elemanı bir girdi vektörünü, d ise istenen çıktı değerini ifade eder. Sınıflandırma problemlerinde d genellikle +1 veya -1 değerlerini alır. Algılayıcı öğrenme kuralı şu şekilde ifade edilir:

1. Ağırlıkların başlangıç değerleri rastsal olarak alınır,
2. Eğitim setinden bir girdi vektörü seçilir,
3. Eğer $y \neq d$ (algılayıcı yanlış cevap verir) ise, $\Delta w_i = \pm \lambda x_i$ denklemi kullanılarak tüm ağırlıklar değiştirilir,
4. İkinci adıma geri dönülür.

Dikkat edilirse bu süreç Hebb Kuralına benzemektedir. Tek fark, eğer ağ doğru cevap verirse hiç bir ağırlıkta değişiklik yapılmaz. Eşik *(θ)* değeri, her zaman $x_0 = 1$ olan **Kukla Girdi Birimi** (Dummy Input Unit) ile nöron arasındaki $w_0=1$ ağırlıklı bağlantıdır ve değiştirilmemektedir.

Eğer algılayıcı yanlış cevap verirse ağırlıklar aşağıdaki denklem kullanılarak değiştirilir.

$$w_i(t+1) = w_i(t) + \Delta w_i(t) = w_i(t) \pm \lambda x_i \qquad \textbf{(4.9)}$$

Algılayıcı Kuramının Yıkılış: 1969 yılında Marvin Minsky ve Seymour Papert tarafından yayınlanan "***Perceptrons: An Introduction to Computational Geometry***" isimli kitap, YSA alanındaki çalışmaların duraklamasına neden olarak gösterilmektedir. Minsky ve Papert, Rosenblatt tarafından önerilen tek katmanlı doğrusal algılayıcının yetenekleri ve kısıtları üzerine, geçerliliğini günümüzde de koruyan yorumlar yanında ayrıntılı analizler sunmuşlardır. Yüzlerce örnek üzerinde çalışarak algılayıcı benzeri cihazların öğrenme, uyum gösterme ve kendi kendini organize etme özelliklerinin basit problemler için geçerli, ancak problemler zorlaştığında geçersiz olduğunu matematik yöntemler kullanarak ispatlamışlardır (Freeman ve Skapura, 1991, s.25) (Akpınar, 1993 s.17).

Minsky ve Papert Algılayıcının sadece **Doğrusal Ayrılabilir** (Linearly Seperable) desenleri sınıflandırabildiğini, buna karşılık bir çok problemin doğrusal ayrılabilir olmadığını ve bu durumun da algılayıcı için önemli sınırlamalara neden olduğunu vurgulamışlardır (Freeman ve Skapura, 1991, s.25) (Akpınar, 1993 s.17). Özellikle mantıksal **XOR** problemlerinin çözümünün olmaması nedeniyle algılayıcı, günümüzde doğrusal olmayan problemlerin çözümünde çok kullanılmamaktadır.

4.1.2. Adaptif Doğrusal Eleman

Adaptif Doğrusal Eleman Projesi: YSA araştırmalarının içerisinde, yapay zekanın bir disiplin olarak doğduğu Dartmouth Konferansına katılan (1956), Stanford Üniversitesinden Bernard Widrow önemli bir yere sahiptir. Widrow ve öğrencileri ilk

olarak 1960 lı yılların başlarında **Adaptif Doğrusal Eleman** (ADALINE) projesini gerçekleştirmişlerdir. Negatif geri besleme yapısının kullanıldığı ADALINE, algılayıcının çözemediği bir çok problemin üstesinden gelmiştir (Akpınar, 1993 s.21).

ADALINE, Widrow ve Hoff tarafından 1959 yılında geliştirilmiştir. Adaptif Doğrusal Eleman'ın (ADAptive LINear Element) kısaltmasıdır. ADALINE genel olarak ADALINE ünitesi olarak adlandırılan tek bir işlemci elemandan oluşmaktadır.

ADALINE, bu bölümün başında anlatılan **Eşik Mantık Birimi** (EMB) ile önemli benzerlikler göstermektedir. ADALINE' da **Kukla Girdi Birimi** veya Yanlı olarak ifade edilen, w_0 ağırlıklı ve girdi değeri (x_0) daima 1'e eşit olan bir bağlantı bulunmaktadır ve çıktı değeri çift kutuplu (Bipolar) olarak belirlenmektedir.

Delta Kuralı: ADALINE modelinde öğrenme, **En Küçük Kareler** (Least Mean Square) yöntemine dayanmaktadır. Öğrenme kuralına **Delta Kuralı** (Delta Bar Delta) da denilmektedir. ADALINE modelinin yapısı algılayıcıya benzemektedir, aralarındaki fark öğrenme kurallarının farklılığından kaynaklanmaktadır. Öğrenme kuralı, ağın çıktısının istenilen çıktı değerine göre hatasını en aza indirecek şekilde ağırlıkların değiştirilmesi temeline dayanmaktadır. Tek katmanlı ve doğrusal aktivasyon fonksiyonuna sahip tek çıktılı bir yapay sinir ağı için çıktı değeri şu şekilde verilebilmektedir;

$$y = \sum_j w_j x_j + \theta \qquad \textbf{(4.10)}$$

Burada x_j, j nöronuna ait girdi; w_j her girdinin ağırlık değeri; θ ise ağ çıktısının 0'dan farklı bir değer almasını sağlayan eşik veya yanlı değeridir ($\theta = w_0 x_0$). Beklenen çıktının d olması durumunda p örneğinin hata değeri ve toplam hata;

$$E^p = d^p - y^p \qquad \textbf{(4.11)}$$

$$E = \frac{1}{2}\sum_p \left(E^p\right)^2 = \frac{1}{2}\sum_p \left(d^p - y^p\right)^2 \qquad \textbf{(4.12)}$$

şeklinde olmaktadır. Burada *p* girdi vektörünün bir elemanı, E_p ise *p* indisli girdinin hata değeridir. En Küçük Kareler kuralı olarak da adlandırılan Delta kuralında, en küçük hata fonksiyonunu verecek ağırlıklar **Dereceli Azalma** (Gradient Descent) yöntemi ile bulunmaktadır. Bu yönteme göre *E*'nin eğimi, *E*'nin her bir ağırlığa bağlı kısmi türevlerinden oluşan bir vektörüdür. *E*'nin eğimi, en hızlı artışın yönünü vermekte, bunun ters yönü ise, hatanın en hızlı azalışını vermektedir.

$$\Delta_p w_j = -\lambda \frac{\partial E^p}{\partial w_j} \qquad \textbf{(4.13)}$$

Bu denklemde λ öğrenme katsayısıdır ve ağırlık vektörünün minimum hata değerine yaklaşması için sürati ve dengeyi sağlamaktadır. Türev zincir kuralı ile alınırsa:

$$\frac{\partial E^p}{\partial w_j} = \frac{\partial E^p}{\partial y^p} \frac{\partial y^p}{\partial w_j} \qquad \textbf{(4.14)}$$

eşitliği sağlanacaktır. Doğrusal birimler:

$$\frac{\partial y^p}{\partial w_j} = x_j \qquad \textbf{(4.15)}$$

$$\frac{\partial E^p}{\partial y^p} = -(d^p - y^p) \qquad \textbf{(4.16)}$$

öyle ki:

$$\Delta_p w_j = -\lambda \delta^p x_j \qquad \textbf{(4.17)}$$

Burada $\delta^p = d^p - y^p$, *p* örneğinin çıktı değeri ile beklenen değeri arasındaki farkı ifade eder ve -E_p değerine eşittir (Kröse ve Smagt, 1996, s.28-29).

Birden fazla ADALINE ünitesinin bir araya gelerek oluşturdukları yapay sinir ağına MADALINE ağı denilmektedir. MADALINE ağı ADALINE ile aynı öğrenme kuralını kullanmaktadır.

4.2. Çok Katmanlı Algılayıcı ve Geri Yayılım Ağı

Algılayıcının kısıtlılığının gösterilmesinden sonra durgunlaşan YSA araştırmaları, 1970'li yıllarda birbirinden bağımsız bir kaç araştırmacının çok katmanlı algılayıcı mimarisi geliştirmesi ve Rumelhart, Hinton ve Williams'ın 1986 yılında geri yayılım algoritmasını geliştirerek XOR problemini çözmesiyle birlikte tekrar hız kazanmıştır (Kröse ve Smagt, 1996, s.33).

Bu çözümün altındaki fikir; ara katmanlardaki birimlerinin hatalarının, çıktı katmanındaki birimlerin hatalarının geriye yayılması ile belirlenmesidir. Bu sebepten dolayı metot **Geriye Yayılım Öğrenme Algoritması** (Back-Propagation Learning Algorithm) olarak anılmaktadır. Bu nedenle **Çok Katmanlı Algılayıcılar** (Multi Layer Perceptron-MLP) **Geri Yayılım Ağı** (Backpropagation Network) olarak da anılmaktadır.

Bu tür ağlar genellikle delta kuralının (Hebb ve algılayıcı kurallarının daha gelişmiş hali), doğrusal olmayan aktivasyon fonksiyonları ve çok katmanlı ağlar için genelleştirilmiş hali olan **Genelleştirilmiş Delta Kuralı** (Generalised Delta Rule) ile eğitilirler. Genelleştirilmiş Delta Kuralı, **En Küçük Kareler Yöntemine** (LMS) dayalı bir öğrenme kuralıdır. Geri yayılım ağına eğitim sırasında hem girdiler hem de o girdilere karşılık gelen çıktılar gösterildiğinden ileri beslemeli danışmanlı bir öğrenme kuralıdır.

Geri yayılım ağı, tek katmanlı algılayıcıların aksine doğrusal olmayan problemlere çözüm üretmeleri nedeniyle günümüzde geniş kullanım alanları bulan en popüler yapay sinir ağıdır. Geri yayılım ağı, algılayıcılar gibi ileri beslemeli olmakla birlikte bir veya daha fazla gizli katmana sahiptirler. Daha geniş uygulama alanlarında kullanılabilmesi için ağda sigmoidal transfer fonksiyonu ile doğrusallıktan kurtarılmış birden fazla çıktı birimi kullanılabilir. Bu sayede çıktı birimleri kesikli adım yapısından sürekli duruma geçirilmiş, yumuşak geçişli karar bölgeleri oluşturulmuş olunur. Yani çıktı birimleri {-1, 0, 1} değerlerinin haricinde [-1,1] aralığında reel sayılar üretebilmektedir.

Yapay sinir ağlarına olan ilginin tekrar artmasına neden olan geri yayılım ağları, başlangıçta akademik çevrelerin ilgisini çekerek üzerinde yapılan çalışmaları

yoğunlaştırmış daha sonra mühendislik problemlerinin hemen hepsine çözüm üretebilecek bir güce sahip olması nedeniyle endüstriyel alanda yoğun bir şekilde kullanılmaya başlanmıştır. Özellikle sınıflandırma, tanıma ve genelleme yapmayı gerektiren problemlerde önemli bir çözüm aracı haline gelmiştir.

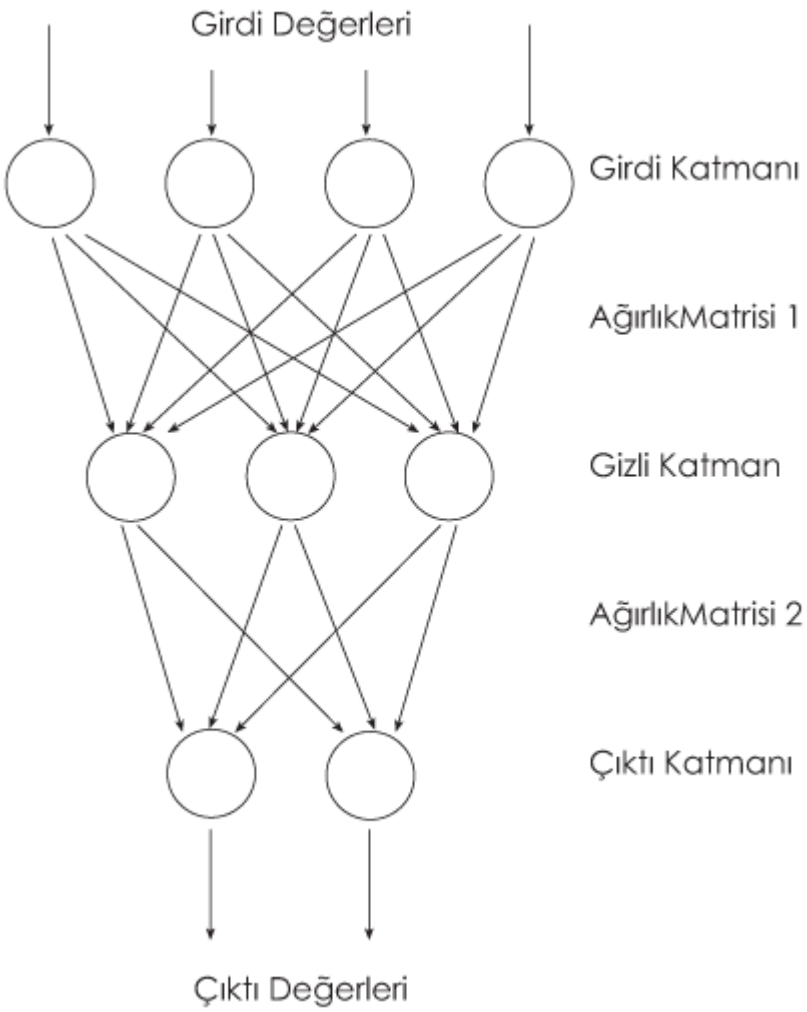
4.2.1. Geri Yayılım Ağlarında Katman Yapısı

Çok katmanlı yapay sinir ağları girdi katmanı, gizli katman ve çıktı katmanı şeklinde en az üç katmandan oluşmaktadır.

Girdi katmanı, dış ortamdan gelen bilgileri alarak ara katmana göndermekle görevlidir. Bu katmanda herhangi bir bilgi işleme olmamaktadır. Geri yayılım ağlarına aynı anda birden fazla girdi gelebilmektedir ancak girdi katmanındaki her işlemci elemanın bir tek girdisi ve bir tek çıktısı olmaktadır. Bu çıktı bir sonraki katmandaki tüm işlemci elemanlara gönderilmektedir. Yani bir katmandaki bir işlemci eleman bir sonraki katmandaki tüm işlemci elemanlara bağlanmaktadır fakat aynı katmandaki işlemci elemanlar arasında herhangi bir bağlantı bulunmamaktadır. Girdi katmanındaki işlemci eleman sayısı uygulanan problemin girdi (Independent Variable) sayısına bağlı olmaktadır.

Gizli katman, girdi katmanından gelen bilgileri işleyerek bir sonraki katmana göndermektedir. Gizli katman olarak da adlandırılan gizli katmanlar bir geri yayılım ağında birden fazla sayıda olabilmektedir. Ayrıca her gizli katmanda birden fazla işlemci eleman bulunabilmektedir. Gizli katman sayısı ve gizli katmanlardaki işlemci eleman sayısı deneme-yanılma yoluyla bulunmaktadır. Gizli katmanda her işlemci eleman bir sonraki katmandaki tüm işlemci elemanlarla bağlantılıdır.

Çıktı katmanı, gizli katmandan gelen bilgileri işleyerek dış ortama göndermektedir. Her geri yayılım ağında bir tek çıktı katmanı bulunmaktadır. Ancak çıktı katmanında birden fazla işlemci eleman bulunabilmektedir. Çıktı katmanındaki işlemci eleman sayısı (Dependent Variable) ise yine uygulanan probleme bağlı olmaktadır.



Şekil 4.4 İki Katmanlı Yapay Sinir Ağı Modeli (Fröhlich, 1996)

Geri yayılım ağlarında bilgiler ağa girdi katmanından sunulmakta, gizli katmanlardan geçerek çıktı katmanına ulaşmakta ve ağa sunulan girdilere karşılık ağın cevabı bu katman tarafından dış ortama iletilmektedir.

İleri beslemeli geri yayılım ağlarında katman sayısı ve katmanlardaki işlemci eleman sayısı ağın başarılı sonuçlar üretebilmesi için önem taşımaktadır. Ancak bu sayıların neler olacağına ilişkin net seçim kriterleri bulunmamaktadır. Bunun yerine uygulamalar sonucunda ortaya çıkmış ve araştırmacılar tarafından kabul görmüş bazı kurallar bulunmaktadır. Bu kurallar şu şekilde özetlenebilmektedir (Anderson ve McNeill, 1992, s.33).

Kural 1: Girdi ve çıktı veriler arasındaki ilişkinin karmaşıklık derecesi arttıkça işlemci eleman sayısı da arttırılmalıdır.

Kural 2: Eğer modellenen problem birçok aşamaya ayrılabiliyorsa, fazla sayıda ara katman kullanılmalıdır. Eğer az sayıda ara katman kullanılırsa ağ öğrenmeyi başaramaz. Gereğinden fazla ara katman kullanılması durumunda ise ağ **ezberler** (Memorization). Bu da ağın, yeni örnekler için genelleme yeteneğini azaltmaktadır.

Kural 3: Yapay sinir ağının eğitilmesinde kullanılan örnek setinin genişliği, ara katmanlardaki işlemci elemanların sayısı için bir üst limit kriteri oluşturmaktadır. Bu üst limiti bulmak için önce eğitim setindeki girdi-çıktı çiftlerinin sayısı bulunmaktadır. Bulunan bu sayı ağdaki toplam girdi ve çıktı işlemci elemanlarının sayısına bölünmektedir. Çıkan sonuç, bir ölçeklendirme katsayısı olarak kullanılabilmektedir. Bu katsayı genellikle 5 ile 10 arasında bir değer olmaktadır. Fazla hata içeren veri setleri için bu ölçeklendirme katsayısı 20 ile 50 arasında değerler alabilmektedir. Diğer taraftan, hemen hemen hiç hata içermemesi durumunda bu katsayı 2 seviyesine kadar düşürülebilmektedir. Bu yöntemle, ölçeklendirme katsayısının ne olacağına bağlı olarak kesin bir sonuca ulaşılamasa da bir fikir edinmek mümkün olmaktadır. Ayrıca, genelleme yeteneğinin kaybolabilmesi ve dolayısıyla yeni veriler tanıtıldığında ağın kullanışsız kalması sonucunu doğurabileceğinden, bir ara katmandaki işlemci eleman sayısının çok fazla olmaması yararlı olacaktır (Baş, 2006).

Geri yayılım ağının **Doğrusal Aktivasyon Fonksiyonu** ile (Linear Activation Function) kullanımı **Doğrusal Regression Modeli**'ni (Linear Regression Model) (Weisberg, 1985) (Myers, 1986), girdi nöronları birden fazla olduğunda **Çoklu Doğrusal Regression Modeli**'ni oluşturur. (Sarle, 1994, s.3)

Girdi ve çıktı katmanındaki nöronların birden fazla olması, çıktı katmanında kullanılan aktivasyon fonksiyonlarının doğrusal olmayan fonksiyonlardan seçilmesi halinde ağ, **Çok Değişkenli Çoklu Regresyon Modeli**'ni oluşturur. (Sarle, 1994, s.5)

Girdi katmanında bir nöron, çıktı katmanında doğrusal fonksiyonlu bir nöron ve gizli katmanda Lojistik Fonksiyon gibi doğrusal olmayan fonksiyonlu birden fazla nöronun olması durumunda ağ **Basit Doğrusal Olmayan Regresyon Modeli**'ni oluşturur . (Sarle, 1994, s.5)

4.2.2. Genişletilmiş Delta Kuralı

İleri beslemeli geri yayılım ağlarında genellikle doğrusal olmayan aktivasyon fonksiyonları kullanıldığı için, Bölüm 4.1.2'de açıklanan ve doğrusal aktivasyon fonksiyonları kullanan delta kuralının genelleştirilmesi gerekmektedir. Bu nedenle bu öğrenme kuralına **Genelleştirilmiş Delta Kuralı** (Extended Delta Bar Delta) denmektedir.

Aktivasyon fonksiyonu toplam girdinin türevlenebilir bir fonksiyonudur ve şu şekilde ifade edilir:

$$y_k^p = \mathcal{F}(s_k^p) \qquad \textbf{(4.18)}$$

y_k^p değeri katman elemanının *p* örneğindeki aktivasyon fonksiyonunun çıktısıdır. Ve net girdi (s_k^p);

$$s_k^p = \sum_j w_{jk} y_j^p + \theta_k \qquad \textbf{(4.19)}$$

şeklinde ifade edilir. w_{jk}, *k* katman elemanının bir diğer katmandaki *j* elemanıyla arasındaki bağlantının ağırlığını ifade eder. θ_k eşik değeri veya yanlı olarak ifade edilir.

Katman sayısı çoğaldığı için değişim katsayısında şu şekilde değişiklik yapılmalıdır:

$$\Delta_p w_{jk} = -\lambda \frac{\partial E^p}{\partial w_{jk}} \qquad \textbf{(4.20)}$$

E_p hatası, aynı şekilde çıktı katmanındaki *o* çıktı biriminin *p* örneğine ait toplam hatası olarak ifade edilir:

$$E^p = \tfrac{1}{2}\sum_{o=1}(d_o^p - y_o^p)^2 \qquad \textbf{(4.21)}$$

burada d_o^p , *p* örneğine bağlı olarak *o* çıktı ünitesinin vermesi istenen çıktı değeridir.

Öncelikle Denklem (4.20)'de ifade edilen değişim katsayısı hesaplanmalıdır.

$$\frac{\partial E^p}{\partial w_{jk}} = \frac{\partial E^p}{\partial s_k^p}\frac{\partial s_k^p}{\partial w_{jk}} \qquad \textbf{(4.22)}$$

Çarpanlar şu şekilde ifade edilir:

$$\delta_k^p = -\frac{\partial E^p}{\partial s_k^p} \qquad \textbf{(4.23)}$$

$$\frac{\partial s_k^p}{\partial w_{jk}} = y_j^p \qquad \textbf{(4.24)}$$

Böylece Bölüm 4.1.2'de açıklanan delta kuralındaki, hata yüzeyindeki **Dereceli Azalma** (Gradient Decent) ile sonuçlanacak değiştirme (Transformation) denklemi elde edilmiş olur.

Ağırlıklar aşağıdaki denklem yardımıyla değiştirilecektir.

$$\Delta_p w_{jk} = -\lambda \delta_k^p y_j^p \qquad \textbf{(4.25)}$$

Buradaki önemli nokta, δ_k^p değerinin ağdaki her *k* birimi için kullanılacak olmasıdır. Burada δ değerlerinin bulunması için hata sinyallerinin ağ boyunca geriye doğru yayılması gerekmektedir.

δ_k^p değerlerinin hesaplanması için, iki faktörün çarpımının kısmi türevi alınır. İlk çarpan, hatadaki değişiminin birimin çıktısı bazındaki fonksiyonu, ikinci çarpan, çıktıdaki değişimin girdideki değişim bazındaki fonksiyonudur. Yani:

$$\delta_k^p = -\frac{\partial E^p}{\partial s_k^p} = -\frac{\partial E^p}{\partial y_k^p}\frac{\partial y_k^p}{\partial s_k^p} \quad \textbf{(4.26)}$$

İkinci çarpanı hesaplarsak:

$$\frac{\partial y_k^p}{\partial s_k^p} = \mathcal{F}'(s_k^p) \quad \textbf{(4.27)}$$

Denklem (4.27), *k* birimindeki net girdiden elde edilen *F* fonksiyonunun bu birimdeki türevidir.

Ağın ağırlıkları değiştirilirken iki durum söz konusudur. Bunlardan ilki, gizli katman ile çıktı katmanı arasındaki ağırlıkların değiştirilmesi; ikincisi ise gizli katmanlar arası veya gizli katman girdi katmanı arasındaki ağırlıkların değiştirilmesidir. Bu iki duruma göre hesaplama işlemi değişiklik göstermektedir.

Yani, ilk çarpanı bulmak için iki durum göz önünde bulundurulur. İlki, *k* biriminin ağın *k=o* çıktı ünitesi olduğudur. Bu durumda tanım gereği E_p şu şekilde bulunur:

$$\frac{\partial E^p}{\partial y_o^p} = -(d_o^p - y_o^p) \quad \textbf{(4.28)}$$

Bu sonuç delta kuralındaki sonucun aynısıdır. Denklem (4.26)'da yerine konursa herhangi bir *o* çıktı birimi için şu denklem elde edilir:

$$\delta_o^p = (d_o^p - y_o^p)\mathcal{F}_o'(s_o^p) \quad \textbf{(4.29)}$$

İkinci olarak *k* birimi bir çıktı birimi değil, ağın *k* = *h* gizli birimidir. Fakat gizli birimin ağın çıktı hatasına katkısının ne olduğu henüz bilinmemektedir. Hata ölçümü gizli katmandan çıktı katmanına yapılan net girdinin bir fonksiyonu şeklinde yazılırsa:

$$\frac{\partial E^p}{\partial y_h^p} = \sum_{o=1}^{N_o} \frac{\partial E^p}{\partial s_o^p} \frac{\partial s_o^p}{\partial y_h^p} = \sum_{o=1}^{N_o} \frac{\partial E^p}{\partial s_o^p} \frac{\partial}{\partial y_h^p} \sum w_{ko} y_j^p$$

(4.30)

$$= \sum_{o=1}^{N_o} \frac{\partial E^p}{\partial s_o^p} w_{ho} = -\sum_{o=1}^{N_o} \delta_o^p w_{ho}$$

ve Denklem (4.26)'da yerine yazılırsa:

$$\delta_h^p = \mathcal{F}'(s_h^p) \sum_{o=1}^{N_o} \delta_o^p w_{ho} \qquad \textbf{(4.31)}$$

denklemi elde edilir.

Denklem (4.29) ve (4.31) ağdaki tüm birimlerin δ değerlerinin hesaplanması için iteratif bir yöntem sunmakta ve Denklem (4.25) yardımıyla ağırlıkların değiştirilmesi için kullanılmaktadır. Bu yöntem, ileri beslemeli ağların doğrusal olmayan birimleri için genelleştirilmiş delta kuralını oluşturur (Kröse ve Smagt, 1996, s.33-16).

Özetle, örnekler ağa iletildikten sonra, aktivasyon fonksiyonlarının çıktıları ağın çıktı elemanlarına yayılırlar. Ağın çıktı değerleri ile istenilen değerler karşılaştırılırlar. Amaç, *o* çıktı elemanının hatasının sıfıra yaklaştırılmasıdır. Her yeni örnek girişinde ağın ağırlıkları delta kuralından da bildiğimiz gibi şu formülle değiştirilmelidir:

$$\Delta w_{ho} = (d_o - y_o) y_h \qquad \textbf{(4.32)}$$

Bu ilk adımdır ama yeterli değildir. Sadece bu formül uygulanarak girdi elemanları ile gizli katmandaki elemanların aralarındaki ağırlıklar değiştirilemez ve bu haliyle ileri beslemeli bir ağ evrensel bir yaklaşım modeli sergileyemez.

Girdi elemanları ile gizli katman elemanları aralarındaki ağırlıklar yine delta kuralı ile değiştirilir. Bu durumda gizli elemanların δ değerlerinin hesaplanması gerekir ve şu

şekilde bulunur: Çıktı elemanının hatası kendisine bağlı olan tüm gizli elemanlara yayılır. Gizli katmanın *h* elemanı, her *o* çıktı elemanının δ değerini alır ve bu δ değerlerinin bağlantı ağırlık değerleri ile çarpımını toplar. Bu işlem şu şekilde formülüze edilir:

$$\delta_h = \sum_o \delta_o w_{ho} \qquad \textbf{(4.33)}$$

Genelleştirilmiş delta kuralı iki adımdan oluşur: ilk adım boyunca *x* girdi değeri ağ boyunca ileri doğru yayılır ve her çıktı elemanı için y_o^p değeri hesaplanır. Bu değer istenen d_o değeri ile karşılaştırılır ve her çıktı elemanı için δ_o^p değeri bulunur. İkinci adım ise hata değerinin ağ boyunca geriye doğru yayılması ve uygun ağırlık değerlerinin hesaplanmasıdır.

4.2.3. Bağlantıların Değiştirilmesi

Bağlantıların ağırlıkları, hatayı girdi olarak alan *k* elemanının δ değeri, bağlantı buyunca bu hata değerini gönderen *j* elemanının çıktı değeri ve öğrenme katsayısının çarpımı yardımıyla değiştirilir.

$$\Delta_p w_{jk} = -\lambda \delta_k^p y_j^p \qquad \textbf{(4.34)}$$

Eğer eleman çıktı elemanıysa hata sinyali (δ):

$$\delta_o^p = (d_o^p - y_o^p)\mathcal{F}_o'(s_o^p) \qquad \textbf{(4.35)}$$

denklemi ile hesaplanır. Eğer *F* fonksiyonu sigmoid fonksiyonsa:

$$y^p = \mathcal{F}(s^p) = \frac{1}{1+e^{-sp}} \qquad \textbf{(4.36)}$$

Türevi:

$$\begin{aligned}\mathcal{F}'(s^p) &= \frac{\partial}{\partial s^p}\frac{1}{1+e^{-s^p}} \\ &= \frac{1}{(1+e^{-s^p})^2}(-e^{-s^p}) \\ &= \frac{1}{(1+e^{-s^p})}\frac{e^{-s^p}}{(1+e^{-s^p})} \\ &= y^p(1-y^p)\end{aligned} \qquad \textbf{(4.37)}$$

Şeklinde hesaplanır.

Böylece çıktı elemanının hata sinyali şu formüle dönüşür:

$$\delta_o^p = (d_o^p - y_o^p)y_o^p(1-y_o^p) \qquad \textbf{(4.38)}$$

Gizli katmandaki elemanın hata sinyali ise, kendisinin bağlı olduğu çıktı elemanlarının hata terimleri ile bağlantılarının çarpımı kullanılarak bulunur. Sigmoid fonksiyon için:

$$\delta_h^p = \mathcal{F}'(s_h^p)\sum_{o=1}^{N_o}\delta_o^p w_{ho} = y_h^p(1-y_h^p)\sum_{o=1}^{N_o}\delta_o^p w_{ho} \qquad \textbf{(4.39)}$$

denklemi ile gösterilir. Sonuçta δ değeri hesaplandıktan sonra, ağırlıklardaki değişiklik, değişim bir önceki ağırlık değişiminin momentumla çarpılmasıyla elde edilen değere bağlıdır:

$$\Delta w_{jk}(t+1) = \lambda \delta_k^p y_j^p + \alpha \Delta w_{jk}(t) \qquad \textbf{(4.40)}$$

Burada *t* iterasyon sayısı, λ öğrenme katsayısı, α ise bir önceki ağırlık değişiminin yeni ağırlık değişimini ne kadar etkileyeceğini belirleyen sabit terimdir (Momentum).

4.2.4. Geri Yayılım Ağlarının Dezavantajları

Geri yayılım ağlarının en büyük sorunu, çalışan sistemin yakınsadığının (Converge) ispatlanamamasıdır. Gizli katmanlarda tam olarak nelerin yaşandığı bilinmediğinden ve sistemin duali alınamadığından ağın doğru çalışıp çalışmadığının matematiksel ispatını yapmak zordur.

Bir diğer sorun, gizli katman ve elemanlarının sayılarının, kullanılacak aktivasyon fonksiyonlarının, ağırlıkların başlangıç değerlerinin, öğrenme katsayısı ve momentum gibi parametrelerin belirlenmesinde kesinlik yoktur. Tüm bu değişkenler deneme yanılma yöntemi ile bulunmaktadır.

Üçüncü bir sorun da ağırlıkların değiştirilmesi sırasında çok fazla işlem yapılmakta olduğundan bu tür ağların eğitimleri uzun sürmektedir. Yine de tüm bu olumsuzluklar bu ağların yaygın bir şekilde kullanılmasını engellememiştir.

Çalışmanın bundan sonraki kısımlarında **Yinelemeli Ağlar** (Recurrent Networks), **Kendinden Organizasyonlu Ağlar** (Self-Organizing Networks) ve bu topolojilerin en bilinen temsilcilerinden bir kaçı tanıtılacaktır.

4.3. Yinelemeli Ağlar

Bölüm 4.1 ve Bölüm 4.2'de ileri beslemeli çok katmanlı ağlarda kullanılan öğrenme algoritmalarında verinin ileri doğru aktığı ifade edilmişti. İleri beslemeli ağlarda, ağ içerisinde veri döngü (Cycle) yapmaz. **Yinelemeli Ağlarda** (Recurrent Networks) gizli işlem elemanları sadece önceki veya sonraki katmanlarla değil, kendisi ile veya aynı katmandaki diğer işlemci elemanlarla da ağırlıklı bağlantılara sahiptir. Bazı ağlarda ise tüm elemanlar birbiri ile bağlantılıdır.

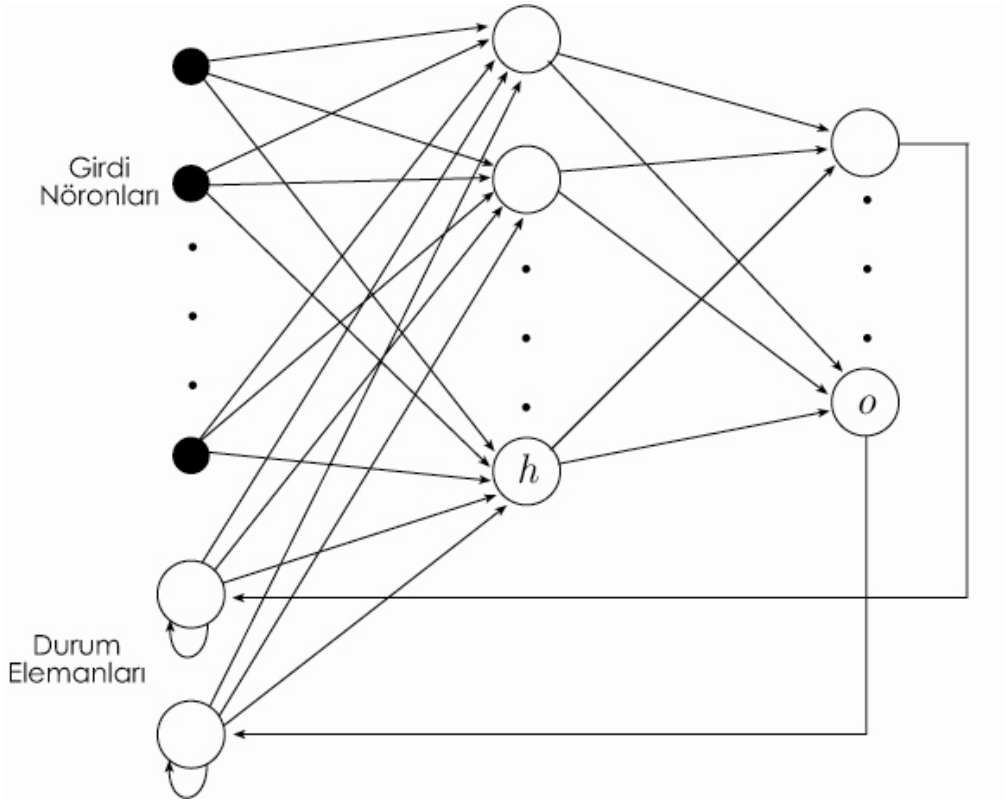
Yinelemeli ağlarda aktivasyon ve hata değerleri sonsuza kadar veya **Kararlı Noktaya** (Stable Point-Attractor) ulaşıncaya kadar yayılabilirler. Aynı katmanlardaki elemanlar birbiri ile bağlantılı olduğundan, gizli katmandaki bir eleman hem dışarıdan gelen hem de kendi katmanındaki elamanlardan gelen verileri girdi olarak kabul edecektir. Yani yinelenen bağlantılar fazladan girdiye yol açacaktır. Bu değerler ağın kendisi tarafından hesaplanmaktadır (Baş, 2006).

Bu ağların dezavantajları, ileri beslemeli ağlardan çok daha fazla girdi boyutuna sahip olmalarıdır. Bu da ağ boyutunu ve hesaplamayı zorlaştıracağı gibi eğitim süresini uzatacaktır. Jordan ve Elman ağları bu probleme çözüm bulmuşlardır.

4.3.1. Jordan Ağları

Jordan ağı, çok katmanlı geri beslemeli (yinelemeli) bir yapay sinir ağıdır. İlk yinelemeli ağlardan biridir. Çok katmanlı algılayıcılara benzer bir yapıda olan Jordan ağlarında, girdi, çıktı ve gizli elemanlara ek olarak **Durum Elemanları** (State Units) adı verilen özel işlemci elemanlar bulunmaktadır. Durum elemanları, çıktı katmanından aldıkları aktivasyon değerlerini bir sonraki iterasyona girdi olarak taşımakla görevli elemanlardır. İstenildiği kadar durum elemanı olabilir.

Durum katmanındaki her işlemci elemanın kendisine bağlantısı bulunmaktadır. Durum elemanlar ile çıktı elemanları arasındaki bağlantıların ağırlık değerleri sabittir ve +1 değerindedir. Dolayısı ile öğrenme, girdi katmanı ile gizli katman arasındaki ve gizli katman ile çıktı katmanı arasındaki bağlantılarda oluşmaktadır. Şekil 4.5 Jordan Ağının şematik gösterimini sunmaktadır.



Şekil 4.5 Jordan Ağı: Çıktı elemanının aktivasyon değeri durum elemanlarına geri besleme yapılıyor.

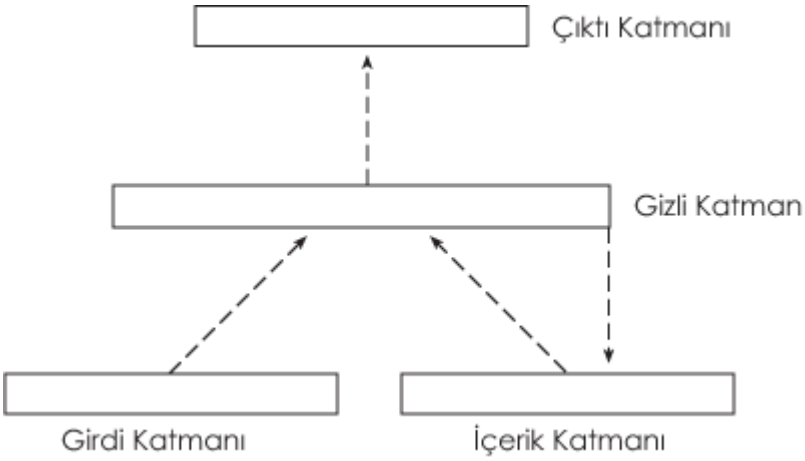
Diğer katmanlardaki işlemci elemanlar, çok katmanlı algılayıcılardaki işlemci elemanlara benzer şekilde çalışırlar. Gizli katman elemanları hem doğrusal hem de doğrusal olmayan aktivasyon fonksiyonlarına sahip olabilmektedir. Çok katmanlı algılayıcıların kullandığı öğrenme kuralları, Jordan ağının eğitiminde de kullanılabilir (Kröse, van der Smagt, 1996).

4.3.2. Elman Ağları

Elman ağı, 1990 yılında Elman tarafından geliştirilmiştir. Elman ağı, girdi katmanı, çıktı katmanı, gizli katmanlar ve bunlara ek olarak **İçerik Katmanından** (Context Layer) oluşmaktadır. İçerik elemanları, gizli katmandan aldıkları aktivasyon değerlerini bir sonraki iterasyona girdi olarak taşırlar.

Elman ağları, Jordan ağına oldukça benzemekle birlikte iki önemli farka sahiptir. Bunlardan ilki geri besleme yaptıkları aktivasyon değerlerini çıktı katmanından değil

gizli katmandan almaları, ikincisi ise içerik elemanlarının kendilerine bağlantılarının bulunmamasıdır. Elman ağlarında da gizli elemanlar ve içerik elemanları arasındaki bağlantı ağırlıkları sabittir ve +1 değerine eşittir.



Şekil 4.6 Elman Ağı: Gizli katmanın aktivasyon değeri, içerik katmanındaki elemanlara geri besleme yapılıyor.

Elman ağının öğrenimi, çok katmanlı algılayıcıların eğitiminde de kullanılan genelleştirilmiş delta öğrenme kuralına göre gerçekleşmektedir. Genelleştirilmiş Delta Kuralı'na göre ağırlıkların değişimi için verilmiş olan denklemler Elman ağı için de geçerlidir. Burada dikkat edilmesi gereken nokta, geri besleme ağırlıklarının değerlerinin sabit olduğu ve değiştirilmeyeceğidir. Yani ağırlıkların değerleri değiştirilirken geri besleme ağırlıklarını dikkate almamak gerekir. Bu ağırlıklar ileri doğru bilgi işlerken içerik elemanlarının girdisini oluşturmada (geri besleme değerlerini girdi olarak içerik elemanlarına taşımada) kullanılırlar.

Ağın ağırlıklarını değiştirirken geri besleme bağlantı ağırlıklarının dikkate alınmaması ve içerik elemanlarının girdi elemanları olarak düşünülmesi durumunda Elman ağı çok katmanlı algılayıcı ile aynı olur (Öztemel, 2003).

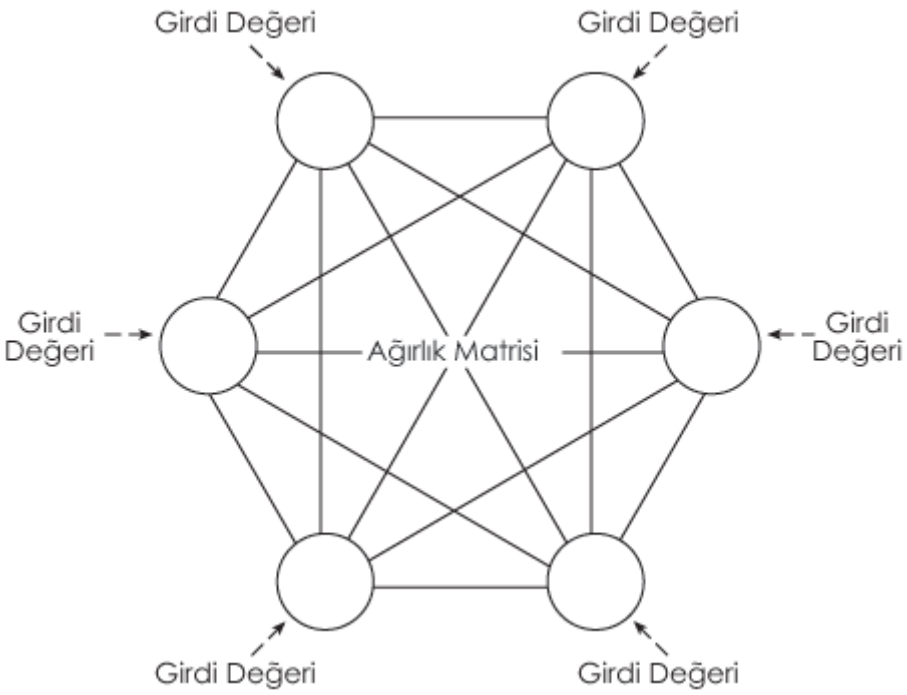
4.3.3. Hopfield Ağı

İlk yinelemeli ağlardan olan **Kendinden İlişkili** (Auto-Associator) Ağlar birbirinden bağımsız iki araştırmacı Anderson ve Kohonen tarafından 1977 yılında tanıtılmıştır. Birbirine bağlı nöronlardan oluşan bir havuz olarak tanımlanmıştır. 1982 yılında, bir fizikçi olan John Hopfield, yinelemeli ağlar fikrinden yola çıkarak Hopfield ağı modelini geliştirmiştir.

Hopfield ağı, beyine benzer bir şekilde belirli hafızaları veya desenleri depolayan basit bir sinir ağıdır. Basit yapısı ve matematiksel tutarlılığı nedeniyle yapay sinir ağları çalışmalarında önemli yeri olan ve tercih edilen bir modeldir. Her probleme uygulanamamasına rağmen, daha karmaşık modelleri anlamak için giriş bilgisi sağlamaktadır (Gülseçen, 1993).

Hopfield ağı tek katmanlı ve geri beslemeli bir yapay sinir ağıdır ve genellikle ikili (0 veya 1) ve çift kutuplu (-1 veya +1) girişleri kabul etmektedir. Hopfield ağında her işlemci eleman diğer işlemci elemanlarla tam bağlantı halindedir ve hem girdi hem de çıktı elemanıdır. Bağlantılar çift yönlüdür (bilgi her iki yönde de akmaktadır) ve simetriktir (Rao, 1995).

Şekil 4.7 Hopfield ağının yapısını göstermektedir.



Şekil 4.7 Hopfield Ağının Yapısı (Fröhlich, 1997)

Ağın bağlantı değerleri bir enerji fonksiyonu olarak saklanır. Her iki yönde akan bilgiye uygulanan ve her bağlantı için hesaplanan bir ağırlık değeri vardır.

Her işlemci eleman diğer işlemci elemanlardan gelen girdilerin toplamını bir eşik değeri ile karşılaştırmakta ve diğer işlemci elemanlara çıktı olarak iletmektedir.

Hopfield ağında örnek desen seçilmekte ve ağın ağırlıklarının başlangıç değerlerini saptamak için kullanılmaktadır. Bu bir kere yapıldıktan sonra herhangi bir desen ağa sunulmakta ve bu da giriş desenine en çok benzeyen desenlerden biriyle sonuçlandırılmaktadır. Çıktı deseni, işlemci elemanların durumlarına bakılarak ağdan okunabilmektedir (Elmas, 2003).

Hopfield ağı, çıktı değerlerinin kesikli ve sürekli oluşlarına göre ikiye ayrılır, bu nedenle **Kesikli** ve **Sürekli** olmak üzere iki tür Hopfield ağından söz edilebilir.

Kesikli Hopfield Ağları, **Birleşik Ağlar veya İlişkili Ağlar** (Associative Networks) sınıfına girmektedir. Birleşik ağlar beynimizin en önemli fonksiyonlarından biri olan

Birleşik Bellek kavramına dayanır. Bir ismin bir kişiyi, bir telefon numarasını, bir başka kişiyi çağrıştırması beynin birleşik bellek özelliğinden kaynaklanmaktadır. Birleşik bellek kavramından hareketle bir grup yapay sinir ağı modelini kapsayan birleşik ağlar ortaya çıkmıştır. Birleşik ağlar kabaca "bir vektör setini başka bir vektör setine bağlayan ağ" olarak tanımlanabilir. **Kesikli Hopfield Ağı** ve **Grossberg Ağı** bu tür ağlardandır (Gülseçen, 1993).

Kesikli ve sürekli Hopfield ağlarının çalışma biçimleri aynıdır ancak kullanılan aktivasyon fonksiyonları farklıdır. Kesikli Hopfield ağları aktivasyon fonksiyonu olarak işaret fonksiyonunu kullanırken sürekli hopfield ağları sigmoid fonksiyonunu kullanmaktadır.

Kesikli Hopfield ağının çalışması şu şekilde açıklanabilir. Bu ağdaki her elemanın iki değeri bulunur. İşlemci elemanların değeri +1 veya -1 olabilmektedir. Eşik değerleri başlangıçta 0 olarak atanmakla birlikte böyle bir zorunluluk yoktur.

Ağın öğrenme aşaması ağırlıkların belirlenmesi aşamasıdır ve bu işlem tek sefer yapılmaktadır. Yani ağın eğitimi bir defada olmaktadır. Ağırlıklar hesaplandıktan sonra sabitlenirler. Ağırlıkların oluşturduğu ağırlık matrisi simetrik bir matristir.

Ağın kullanılabilmesi için **Kararlı** (Stable) hale gelmiş olması gerekir. Bunun için ağa daha önce görmediği yani eğitim setinde olmayan bir örnek gösterilir. Bu örnek eksik bilgiler içerebilir. Ağın görevi bu eksiklikleri gidermek ve örneğin tamamını hafızadan bulmaktır. Bunun için örnek ağa sunulmakta ve ağ durağan hale gelene kadar iterasyonlar yapılmaktadır. Ağ durağan hale geldikten sonra ürettiği çıktı, ağın kendisine gösterilen girdiye cevabı olarak görülmektedir (Baş, 2006).

Ağın durağan hale gelebilmesi için enerji fonksiyonunun değerinin minimize edilmesi gerekmektedir. Ağ çalışırken bu enerji fonksiyonu ya azalır ya da değişmez. Dolayısıyla zaman içinde ağın minimum hata düzeyine ulaşması (kararlı hale gelmesi) her durumda mümkün olmaktadır.

Hopfield ağlarının ana kullanım amacı İlişkili Hafızadır (Associative Meomory). Bunun yanında Hopfield ağları optimizasyon problemlerinde ve veri ilişkilendirmede

yaygın olarak kullanılmaktadır. Hopfield ağlarının en önemli uygulamalarından biri, çözümü çok zor olan gezgin satıcı problemidir. Bu problemde bir satıcı N adet şehre gitmek zorundadır. Bir şehre bir defa uğramak koşulu ile en kısa zamanda bütün şehirleri gezebilmesi için izlemesi gereken rotanın bulunması istenmektedir.

4.3.4. Boltzmann Makinesi

Boltzmann Makinesi, 1985 yılında Ackley, Hinton, ve Sejnowski tarafından tanımlanmıştır. **Cauchy Makinesi** olarak da adlandırılmaktadır. Olasılık ve istatistiksel temelli ağlara girmektedirler. İki önemli farkla Hopfield ağlarına benzemektedir. Gizli katmana sahiptir ve öğrenme kuralı deterministik değil, stokastiktir.

Ağırlıklar Hopfield ağında olduğu gibi simetriktir. Ağın operasyonu fizikteki, bir maddenin donma noktasına kadar çok yavaş soğutulması anlamına gelen **Yavaş Soğutma** (Annealing) prensibine göre çalışır. Fizikte bu işlem maddenin kristalleşmesi ve minimum enerji seviyesine inmesi ile sonuçlanır.

Bu ağlarda kullanılan eğitim kuralı olasılık tabanlı **Benzetmeli Soğutma** (Simulated Annealing) olarak adlandırılmaktadır. Gizli katmanların bağlantıları asenkron ve stokastik olarak değiştirilirler. Ağırlıkların değiştirilmesi sırasında yapay bir sıcaklık kavramı ve olasılık kavramı kullanılmaktadır. Sıcaklık değeri başlangıçta yüksek bir değer olarak atanır, iterasyonlarla düşürülmeye, sonuçta sıfıra indirilmeye çalışılır. Sıfıra inememesi durumunda sıcaklık yükseltilerek iterasyon sayısı arttırılır. Bu sayede yerel minimuma inmekten kaçınılmış olur.

Hopfield ağında olduğu gibi ağ bir kere öğrendikten sonra eksik desenleri tanımlayabilmekte ve tamamlayabilmekedir. Veri ilişkilendirmede, optimizasyon problemlerinde ve robotik sistemlerde kontrol amaçlı kullanılmaktadır.

4.4. Özörgütlemeli Ağlar

Bu bölümde, **Yarışmacı Öğrenme** (Competetive Learning) kuralını uygulayan ağlar tanıtılmaktadır. Danışmansız öğrenen bu tür ağlarda nöronlar arasında bir yarışma söz konusudur. Yarışmacı öğrenme kuralı ilk kez Rumelhart ve Zipser tarafından ortaya atılmıştır. Bu tür ağların en iyi örnekleri, Kohonen tarafından ortaya atılan **Özörgütlemeli Özellik Haritası** (Self-Organizing Map), Carpenter ve Grossberg tarafından kullanılan **ART** (Adaptive Resonance Theory) ağları ve **LVQ** (Learning Vector Quantisation) ağlarıdır.

Özörgütlemeli ağlar, kümeleme işlemlerinde, sürekli fonksiyonların boyutlarının düşürülmesi veya kesikli (Discrete) hale getirilmesinde ve veri içindeki özelliklerin çıkarılmasında kullanılmaktadırlar.

4.4.1. Kohonen Özörgütlemeli Özellik Haritası

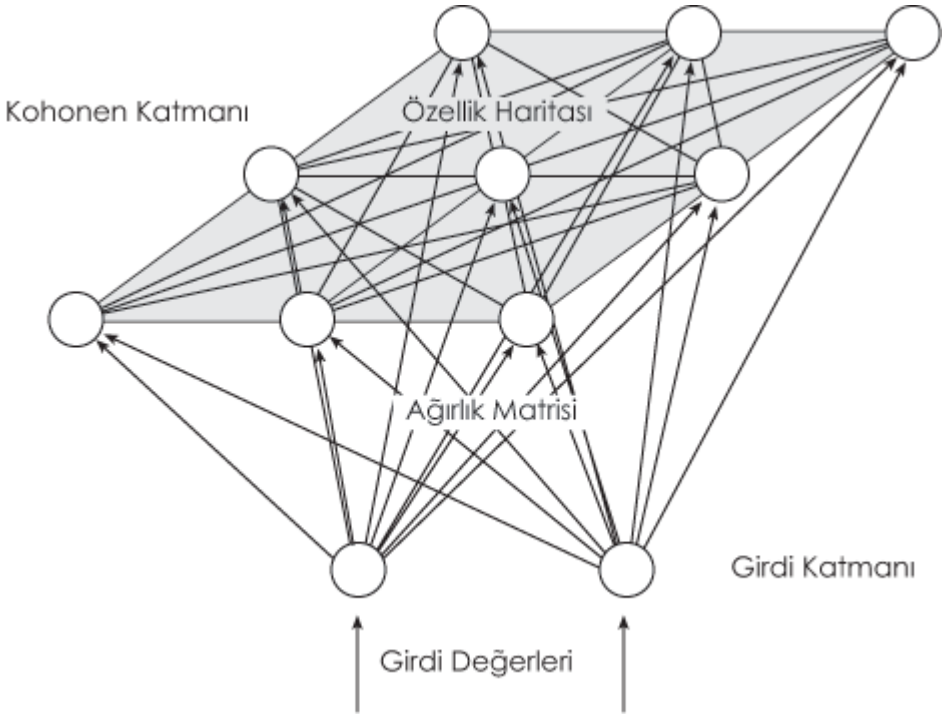
SOM (Self Organization Feature Map Network-Özörgütlemeli Özellik Haritası) olarak da adlandırılan Kohonen ağı, beynin Neocortex katmanında yaygın olan duyusal haritalardan esinlenilerek 1972 yılında Kohonen tarafından geliştirilmiştir.

Genel olarak sınıflandırma ve kümeleme yapmak için kullanılan Kohonen ağının, girdi vektörlerinin dağılımını öğrenebilme ve girdi vektörlerini sınıflandırma yetenekleri oldukça yüksektir (Baş, 2006).

Kohonen ağlarında sıraları korunmak şartı ile girdi vektörlerinin iki boyutlu düzleme izdüşümleri alınır. Birbirine yakın vektörler iki boyutlu Kohonen katmanındaki birbirine yakın işlemci elemanlara haritalanırlar (mapping). Bu ağlarda işlemci elemanlar girdi vektörlerinin iki boyutlu izdüşüm haritalarını oluştururlar. Kohonen ağının asıl amacı, yüksek dereceden girdi uzaylarının hiyerarşik yapılarını ve topolojilerini iki boyutlu olarak gösterebilmektir.

Kohonen aynı zamanda optimizasyon problemlerinde, bağlantı ağırlıklarını minimum enerji desenine indirgemek amacıyla kullanılırlar. Bu tür ağların diğer ağlardan en önemli farkı, eğitim sırasında danışmana ihtiyaç duymamalarıdır. Bu yeteneklerinden dolayı **Özörgütlemeli Harita** (Self-Organizing Map, SOM) olarak da

anılırlar. Eğer tahmin veya sınıflandırma amacıyla diğer ağ topolojileri ile birlikte kullanılırlarsa, önce özörgütlemeli ağlar kullanılarak ağın kendi başına öğrenmesi sağlanır, daha sonra danışmanlı öğrenme konumunda tahmin için kullanılan diğer ağların eğitimine geçilir. Şekil 4.8'de özörgütlemeli Kohonen ağı örneklendirilmiştir.



Şekil 4.8 Özörgütlemeli Ağ Yapısı.

Özörgütlemeli harita ağı, tipik olarak iki katmandan oluşur. Girdi katmanı ve iki boyutlu Kohonen çıktı katmanı. Girdi katmanı, Kohonen katmanındaki tüm işlemci elemanlarla bağlantılıdır. Yarışmacı öğrenmeyi kullanan Kohonen ağında kazanan işlemci eleman 1 diğer elemanlar ise 0 değerini alır. Bu stratejiye **Kazanan Hepsini Alır** (Winner-Takes-All) stratejisi denir.

Eğitim sırasında hem yarışmayı kazanan işlemci elemanın hem de komşusu olan işlemci elemanların ağırlıkları değiştirilir. Yakın işlemci elemanlar destekleyiciyken (Excitatory), uzak elemanlar engelleyicidir (Inhibitory).

Kohonen ağının eğitiminde, herhangi bir *t* zamanında örnek setinden herhangi bir örnek ağa gösterilir. Ağdaki ağırlık değerleri başlangıçta rasgele olarak seçilir. Girdi vektörü ve ağırlık vektörü normalize edilmiş olmalıdır. Çıktı elemanlarından kazanan işlemci elemanı bulmak için iki yöntem bulunmaktadır.

Bunlardan ilkinde, her elemanın çıktısı girdilerin ağırlıklı toplamı ile bulunur ($y_i = \sum w_i x_i$). Bu çıktı değerlerinden en yüksek değere sahip olan işlemci eleman yarışmayı kazanır. İkinci yöntemde ise, Öklit mesafesi kullanılarak elde edilen ($d_i = \|X - W_i\|$), girdi vektörüne en yakın ağırlık vektörüne sahip işlemci eleman kazanan elemandır. Her çıktı elemanı için bu mesafeler hesaplanmakta ve en küçük mesafe değerine sahip işlemci eleman kazanan eleman olarak belirlenmektedir.

Kazanan işlemci elemanın belirlenmesinin ardından bu elemanın ve komşularının ağırlıkları değiştirilir. Yani kazanan işlemci elemanın çevresindeki (aynı bölge içindeki) elemanlar kazanan elemanla aynı cevabı vermesi için desteklenir. Eğitim esnasında öğrenme katsayısı sürekli olarak düşürülür ve komşuluk alanı sürekli daraltılır. Yani geniş bölgelerden başlanır, giderek bölgeler daraltılır.

Kazanan işlemci elemanın, onunla birlikte ağırlıklarının değiştirileceği, komşularını belirlemek için iki yöntem bulunmaktadır. Bunların ilki, kazanan işlemci elemanın etrafındaki elemanları bir kare/dikdörtgen içine almak ve bunun içinde kalanların ağırlıklarını değiştirmektir. İkinci yöntem ise, kazanan işlemci elemanın etrafındaki elemanlar bir çokgen içine alınarak içinde bulunan elemanların ağırlıklarının değiştirilmesidir.

Kohonen ağları, ses ve yazı tanıma, eş zamanlı çeviri gibi işlemlerde oldukça başarılıdır. Renk, desen ayrımlarını yapabilir ve görünen, hareket eden nesneleri ayrıştırabilirler. Bu özellikleri sayesinde robotik sistemlerde ve sensör uygulamalarında da kullanılmaktadırlar.

4.4.2. Doğrusal Vektör Parçalama/Niceleme Modeli (LVQ)

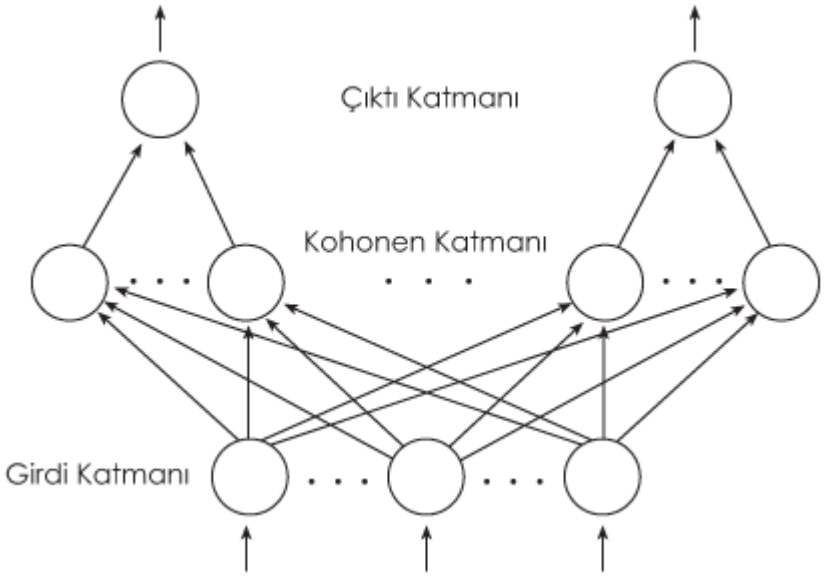
LVQ (Learning Vector Quantization) ağı 1984 yılında Tuevo Kohonen tarafından geliştirilmiştir. LVQ ağlarının temeli, Bölüm 4.4.1'de bahsedilen, Kohonen'in daha önce geliştirdiği SOM (Self-Organizing Maps) modelindeki Kohonen katmanıdır.

Sınıflandırma problemi için geliştirilmiş olan LVQ modelinin temel felsefesi, *n* boyutlu bir vektörü bir vektörler setine haritalamaktır. Çok hızlı sonuçlar üretmeleri ve performanslarının çok yüksek olması nedeni ile tercih edilen ağlardandır.

LVQ ağının öğreniminden kasıt, girdi vektörünün hangi vektör seti tarafından temsil edildiğinin bulunmasıdır. Bu vektör setine referans vektörleri denir. LVQ ağının görevi, öğrenme yolu ile bu referans vektörlerini belirlemektir. Yani, girdi vektörlerinin üyesi olabilecekleri vektör sınıfını belirlemektir (Öztemel, 2003).

LVQ ağları üç katmandan oluşmaktadır. İlk katman olan girdi katmanında bilgi işleme olmamaktadır. Gelen bilgiler girdi katmanını oluştururlar. İkinci katman, Kohonen katmanı da denilen ara katmandır. Bu katmanda SOM'daki gibi girdi setine en yakın olan ağırlık vektörü belirlenir. Bu katmandaki her eleman bir referans vektörünü gösterir.

Girdi vektörü, girdi katmanı ile Kohonen katmanı arasındaki ağırlıkların oluşturduğu referans vektörlerine haritalanmaktadır. Üçüncü katman olan çıktı katmanında ise girdinin ait olduğu sınıf belirlenir (Baş, 2006). Şekil 4.9, LVQ ağının şematik yapısını örneklemektedir.



Şekil 4.9 LVQ Ağının Yapısı

LVQ ağları, girdi ve Kohonen katmanları arasında tam bağlantılı, Kohonen ve çıktı katmanları arasında ise kısmi bağlantılı olmaktadır. Yani girdi katmanındaki her işlemci eleman Kohonen katmanındaki tüm işlemci elemanlarla bağlantılıdır. Kohonen katmanındaki işlemci elemanlar ise çıktı katmanındaki bir tek işlemci elemanla bağlantılıdır. Kohonen katmanı ile çıktı katmanı arasındaki ağırlıklar sabit olup 1'e eşittir ve bu ağırlıklar değişmez.

Ağın eğitimi sadece Kohonen katmanı ile girdi katmanı arasındaki ağırlıkların, yani referans vektörlerinin değerlerinin değiştirilmesi ile gerçekleştirilir. Bu değişiklikler sayesinde girdileri doğru sınıflara ayıracak referans vektörleri belirlenmektedir. Ağ eğitilirken her yinelemede ağın ürettiği çıktının değeri yerine sadece doğru olup olmadığı belirlenir. Sadece girdi vektörüne yakın olan vektörün (kazanan vektör) değerleri (ağın bu vektöre ait ağırlıkları) değiştirilir (Kröse, van der Smagt, 1996).

Hem Kohonen katmanındaki hem de çıktı katmanındaki her işlemci elemanın çıktıları ikili (Binary) değerler almaktadır ve sadece bir işlemci elemanın çıktı değeri 1 diğerleri ise 0 olmaktadır. Çıktı değerinin 1 olması girdinin, ilgili çıktının temsil ettiği sınıfa ait olduğunu göstermektedir.

LVQ ağlarının eğitiminde **Niceleme** (Quantization) algoritması kullanılır. Bir ağının eğitilmesindeki amaç her iterasyonda girdi vektörüne en yakın referans vektörünü bulmak, yani, **Küme Merkezlerini** (Cluster Center) ayarlamak ve eğitim setindeki tüm girdi vektörlerinin niceleme hatalarını minimize etmektir. Referans vektörleri daha önce de belirtildiği gibi Kohonen katmanındaki işlemci elemanları girdi katmanındaki işlemci elemanlara bağlayan ağırlık değerleridir.

Öğrenme esnasında sadece referans vektörlerinin ağırlık değerleri değiştirilmektedir. Bu işlem Kohonen öğrenme kuralı kullanılarak yapılmaktadır. Kohonen öğrenme kuralı, Bölüm 4.4.1'de de anlatıldığı gibi Kohonen katmanındaki işlemci elemanların birbirleriyle yarışması ilkesine dayanmaktadır. Yarışma kriteri, girdi vektörü ile ağırlık vektörleri (referans vektörler) arasındaki Öklid (Euclid) mesafesine bağlıdır. Referans vektörü girdi vektörüne en yakın işlemci eleman yarışmayı kazanır. Kazanan işlemci eleman için iki durum söz konusu olmaktadır (Cozart, 1996).

İlk durumda kazanan işlemci eleman doğru sınıfın bir üyesidir. Bu durumda ilgili ağırlıklar girdi vektörüne biraz daha yaklaştırılmaktadır. Bu, aynı örnek ağa tekrar gösterildiğinde yine aynı işlemci elemanın kazanması için yapılır. Bu durumda ağırlıkların değiştirilmesi gerçekleştirilir. Öğrenme katsayısı zaman içinde 0 değerini alacak şekilde monoton olarak azaltılır. Bunun nedeni, girdi vektörünün referans vektörüne çok yaklaştığında durması ve aksi yönde tekrar uzaklaşmamasıdır. Aksi takdirde ters yönde tekrar uzaklaşma meydana gelecektir.

İkinci durumunda ise, kazanan işlemci eleman yanlış sınıftandır. Bu durumda ağırlık vektörü girdi vektöründen uzaklaştırılmalıdır. Bunun amacı, bir daha aynı örnek geldiğinde aynı işlemci elemanın kazanmamasıdır. Bu durumda ağırlıklar değiştirilir. Öğrenme katsayısının zaman içinde azalması burada da geçerli olmaktadır.

Kohonen katmanı ile çıktı katmanı arasındaki ağırlıklar eğitim sırasında değiştirilmemektedir. Kohonen katmanındaki işlemci elemanların çıktıları, bu işlemci elemanları çıktı katmanına bağlayan ağırlık değerleri ile çarpılarak ağın çıktısı hesaplanmaktadır. Bu Kohonen katmanında yarışmayı kazanan işlemci elemana bağlı olan çıktı elemanın değerinin 1, diğerlerinin değerinin 0 olması anlamına

gelmektedir. Ağın çıktıları belirlendikten sonra çıktının doğru sınıflandırılıp sınıflandırılmadığı sorgulanır. Bu sorunun cevabına göre Kohonen katmanındaki yarışmayı kazanan işlemci elemanı girdi katmanına bağlayan ağırlıklar değiştirilmektedir. Bu nedenle LVQ ağları Takviyeli Öğrenme sınıfındadır.

Eğitim setindeki tüm örnekler doğru sınıflandırılıncaya kadar bu işlemler tekrarlanır. Hepsi doğru sınıflandırılınca öğrenme gerçekleştirilmiş olur. Anlatılan bu model standart bir LVQ modelidir.

LVQ modelinin bazı dezavantajları bulunmaktadır. Bunlardan ilki öğrenme katsayısının zamanında 0 değerini almaması durumunda ağın doğru ağırlık değerlerinden uzaklaşmasıdır. Ayrıca bazı problemlerde sürekli aynı referans vektörü yarışmayı kazanmakta olduğundan ağın esnekliği ortadan kalmaktadır.

Diğer dezavantaj ise sınıflandırmayı yaparken iki sınıfın tam ortasında veya sınırlara çok yakın bulunan vektörlerin hangi sınıfa gireceklerinin belirlenememesidir. Bu sorunları ortadan kaldırmak için LVQ2, Ceza Mekanizmalı LVQ ve LVQ-X modelleri geliştirilmiştir.

LVQ2 Ağı: yine Kohonen tarafından geliştirilmiştir. LVQ2, standart LVQ modelinin uygulanmasından sonra elde edilen çözümün iyileştirilmesi için kullanılır. Sınır değerlerdeki vektörlerin yanlış sınıflandırılmasını önlemeye çalışır. LVQ'nun aksine, aynı anda iki referans vektörünün ağırlıkları değiştirilir. Bu vektörlerin ağırlıklarının değiştirilmesi için iki koşulun sağlanması gerekmektedir.

1. Girdi vektörüne en yakın iki vektörden, birincisi (en yakın olan) yanlış, ikincisi doğru sınıftadır.
2. Girdi vektörü, bu iki vektörün arasında merkezi olarak belirlenmiş bir aralık içerisinde kalmaktadır.

Bu durumda iki vektörün de ağırlıkları değiştirilir.

Ceza Mekanizmalı LVQ: Standart LVQ ağlarında bazı ağırlık vektörleri sürekli kazanmakta ve bu da ağın esnekliğini bozmakta, diğer ağırlık vektörleri referans vektörü olma niteliklerini kaybetmektedir. Belirlenen her ağırlık vektörü girdinin bir yönünü göstermesi gerekirken sadece bir tanesi tüm sorumluluğu üstlenmektedir.

Bu problemden kurtulmak için 1988 yılında DeSieno tarafından Ceza Mekanizmalı LVQ modeli geliştirilmiştir. Bu modelde sürekli kazanan ağırlık vektörü cezalandırılmakta ve peş peşe kazanması engellenmektedir. Sürekli kazanan vektöre kaç defa kazandığı ile ilgili olarak uzaklığına bir değer eklenir, yani uzaklaştırılır. Bu sayede diğer vektörlere de kazanma şansı tanınmaktadır.

LVQ-X Modeli: LVQ2 modelinde olduğu gibi aynı anda iki ağırlık vektörünün değerlerini değiştirmekte, bu sayede hem öğrenme hızı hem de ağın genelleme yeteneği arttırılmaktadır. Bu öğrenme kuralı Öztemel tarafından geliştirilmiştir. Her iterasyonda mutlak kazanan ve yerel kazanan olmak üzere iki işlemci eleman seçilir. Mutlak kazanan girdi vektörüne en yakın ağırlık vektörü, yerel kazanan ise, doğru sınıf içerisinde girdi vektörüne en yakın ağırlık vektörüdür.

Eğer mutlak kazanan işlemci eleman doğru sınıf içerisinde değilse onun ağırlık vektörü o girdi vektöründen uzaklaştırılır. Aynı zamanda doğru sınıf içerisindeki en yakın olan işlemci elemanın (yerel kazanan) ağırlık vektörü de girdi vektörüne yaklaştırılır. Bu doğru işlemci elemana daha sonraki iterasyonlarda kazanma şansı vermektedir.

Eğer mutlak ve yerel kazanan aynı işlemci eleman ise o zaman tek bir ağırlık vektörü değiştirilmekte ve bu ağırlık vektörü girdi vektörüne yaklaştırılmaktadır. (Öztemel, 2003)

4.4.3. Uyarlamalı Rezonans Teorisi (ART)

Diğer özörgütlemeli ağlardan farklı olarak sadece ileri beslemeli değil, hem ileri hem geri beslemeli bir ağ olan Uyarlamalı Rezonans Teorisi (Adaptive Resonance Theory, ART) ağları Grossberg'in 1976 yılında beynin fonksiyonlarını açıklamaya yönelik araştırmalarının sonucunda ortaya çıkmıştır. Bu model, biyolojik sisteme ait şu üç temel üzerine kurulmuştur (Kröse, van der Smagt, 1996).

1. Normalizasyon: Biyolojik sistemler bulundukları çevredeki değişimlere duyarlıdır ve bu değişikliklere uyum sağlayabilme özelliklerine sahiptir. Örnek olarak, insanların fazla gürültülü bir ortama ya da karanlık bir ortama uyum sağlaması, çevresindeki olayları normalize ettiğini göstermektedir.

2. Ayrıştırabilme: İnsanların fark edilmesi zor olan ayrıntıları ayrıştırabilme yeteneği bazen hayat kurtaran bir özelliktir. Örneğin insan, dinlenen bir kaplanla saldırmak üzere olan bir kaplan arasındaki farkı kolayca anlayabilecek bir sisteme sahiptir.

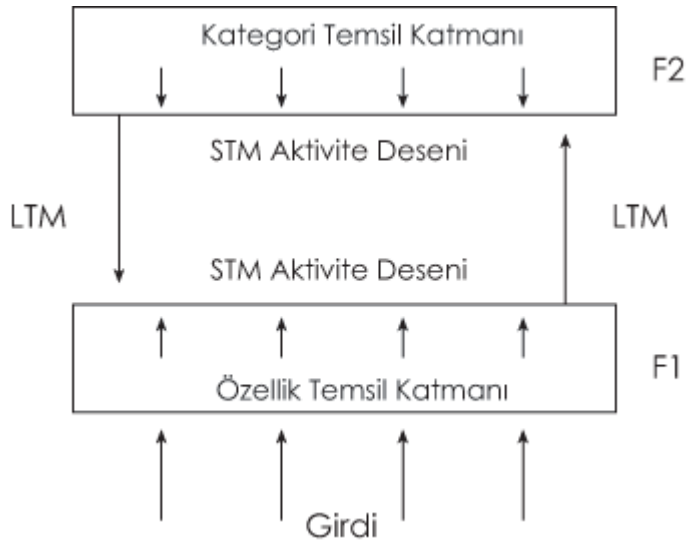
3. Ayrıntıların Saklandığı Kısa Dönemli Hafıza: Belirlenen farklılıklar ve çevresel değişimler davranışlara dönüşmeden önce hafızada saklanmaktadır. Bilgilerin geçici olarak tutulduğu ve zaman içinde yok olarak yerine yeni bilgilerin saklandığı yer **Kısa Dönemli Hafızadır** (Short Term Memory, STM). Ancak sürekli aynı şeylerin tekrar edilmesi sonucunda bilgiler kısa dönemli hafıza yerine **Uzun Dönemli Hafızada** (Long Term Memory, LTM) saklanabilmektedir. Uzun dönemli hafıza bilgilerin sürekli tutulduğu ve kolay kolay unutulmadığı hafıza türüdür.

Grossberg ve arkadaşları beynin bu özelliklerinden yola çıkarak, beynin kullandığı sezgisel yaklaşımları matematik modele dönüştürmüşler ve ART ağlarını oluşturmuşlardır.

ART, adını öğrenme ve hatırlama arasındaki karşılıklı etkileşimi gerçekleştiren yöntemden yani rezonanstan almıştır. ART yapay sinir ağlarında işlemci elemanların çıktıları katmanlar arasında sürekli olarak ileri geri hareket etmektedir. Bu esnada eğer örnek girdi belirlenmiş bir sınıfa uyuyorsa ağ kararlı hale gelmekte ve ART ağı rezonanstadır denilmektedir (Gülseçen, 1993).

ART ağları, danışmansız öğrenmeye dayalı yapay sinir ağlarıdır. Sınıflandırma problemlerinin çözümünde kullanılmak üzere geliştirilmiş olan ART ağları, daha sonra geliştirilen danışmansız öğrenme ağlarının ortaya çıkmalarına da temel oluşturmuştur. ART ağları gerçek zamanda çalışabilir ve çevrimiçi öğrenebilirler.

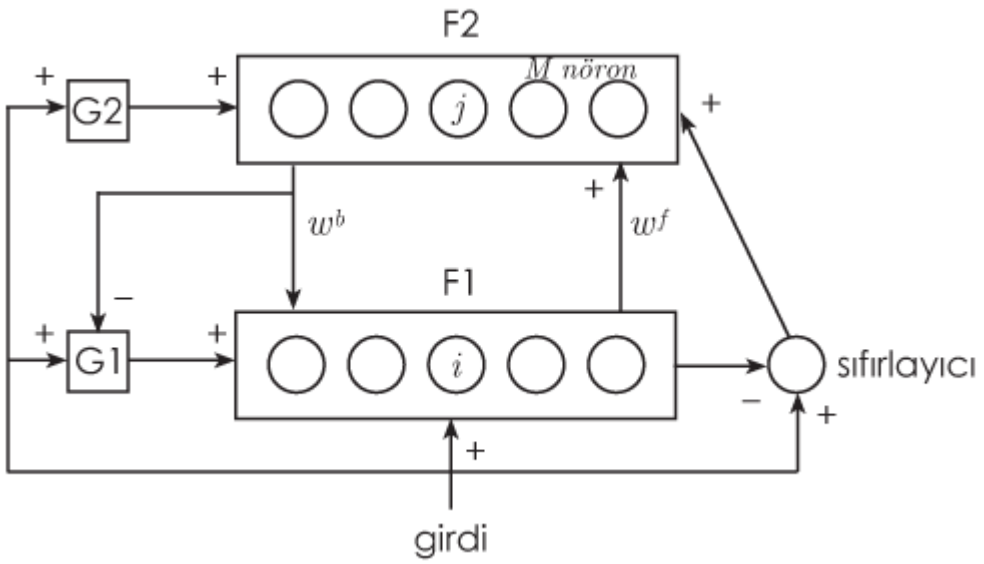
ART ağları, genel olarak iki katmandan oluşur. Bu katmanlar F1 ve F2 olarak isimlendirilmiştir. F1 katmanı girdinin özelliklerini, F2 katmanı kategorileri (ayrıştırılmış sınıfları) gösterir. Bu iki katman birbirlerine LTM ile bağlanırlar. Girdi bilgileri F1 katmanından alınarak, F2 katmanında sınıflandırma yapılmaktadır. Şekil 4.10, ART ağının şematik yapısını örneklemektedir.



Şekil 4.10 ART Ağı Yapısı

ART ağlarında girdiler sınıflandırılmadan önce, özellikleri incelenerek **Özellik Temsil Katmanının** (F1) katmanının aktivasyonu belirlenir. **Kategori Temsil Katmanında** (F2) LTM bağlantılarının ağırlıklarındaki beklentiler kategori desenlerine çevrilir. Yapılan sınıflandırma, F2'den F1'e giden LTM bağlantılarının ağırlıklarındaki beklenti değerleri ile karşılaştırılır ve ağın beklentilerine uygun olup olmadığı görülür. Eğer örnek belirlenmiş bir sınıfa uyuyorsa o kategoriye ait olarak kabul edilir. Aksi takdirde ya yeni bir sınıf oluşturulur ya da reddedilir.

ART ağlarının çalışması, aşağıdan yukarı (F1'den F2'ye) bilgi işleme ve yukarıdan aşağı (F2'den F1'e) bilgi işleme olmak üzere iki şekilde olmaktadır. Aşağıdan yukarı bilgi işleme durumunda bir girdi deseni ağa gösterilir. Bu desen hem F1 katmanında STM'de *X* aktivite desenini oluşturur hem de oryantasyon sistemini veya diğer bir deyişle **Sıfırlama Modülünü** (Reset Modülü) aktif etmek üzere bir işaret (Signal) gönderir. Benzer şekilde oluşturulan *X* deseni hem sıfırlama modülüne bir engelleyici (Inhibitory) işaret göndermekte hem de F1 katmanından bir çıktı deseni oluşturmaktadır. Bu çıktı deseni F2 katmanına giden bir girdi desenine dönüştürülür. Bu girdi deseni ise F2 katmanının çıktısı olan deseni oluşturur. Bu aynı zamanda ağın çıktısıdır. Bu şekilde aşağıdan yukarı (F1 katmanından F2 katmanına) bilgi işleme tamamlanmış olmaktadır. Şekil 4.11'de ART ağının çalışma şekli gösterilmiştir.



Şekil 4.4.11 ART Ağının Çalışma Şekli

Yukarıdan aşağı bilgi işleme de benzer şekilde gerçekleşmektedir. Bu durumda, F2 katmanında oluşturulan çıktı deseni yukarıdan aşağı bir sinyal gönderir. Bu sinyal daha sonra beklenti desenine dönüştürülür. Aynı zamanda kontrol faktörü (kazanç, Gain) için engelleyici (Inhibitory) bir işaret gönderir. Bundan sonra beklenti deseninin girdi deseni ile eşlenip eşlenemeyeceğine bakılır. Eğer böyle bir eşleşme mümkün değilse F1 katmanında yeni bir STM deseni oluşturulur. Bu desen oryantasyon sistemindeki engelleyici işaretin etkisini azaltmaktadır.

Oluşturulan yeni sinyal oryantasyon sistemindeki engelleyici işaretin etkisini azaltarak sıfırlama modülünün F2 katmanına bir sinyal göndermesini sağlar. Bu işaret F2 katmanında yeni desen oluşturur. Böylece ağa gösterilen girdi deseni için doğru sınıfı gösteren yeni bir çıktı üretilir.

Eğer üretilen desen ile girdi deseni eşleşirse o zaman sadece yukarıdan aşağı o girdinin sınıfını gösteren ağırlıklar değiştirilir. Bu değiştirme öğrenme kuralına göre gerçekleştirilmektedir.

1976 yılından bugüne dek çok sayıda ART ağları tanımlanmıştır. Bunlar arasında ART1, ART2, ART3, ARTMAP, Fuzzy ART gibi ağları saymak mümkündür. Bu ağların hepsi aynı temel felsefeye dayanmakta ancak öğrenme kurallarında çok az farklılıklar göstermektedir

ART ağlarının diğer yapay sinir ağlarından farkları: ART ağlarının bilinen diğer ağlardan temel farklılıklarını Grosberg şu şekilde özetlemiştir: (Öztemel, 2003)

- ART ağları gerçek zamanlı olarak oldukça hızlı ve kararlı bir şekilde öğrenme yeteneklerine sahiptirler. Diğer bir çok ağda olmayan bu yetenek sayesinde ART ağları, donanımla desteklenerek gerçek zamanlı öğrenebilen bilgisayarlarda kullanılmaktadırlar.
- Diğer ağların çoğu çevrimdışı öğrenme yeteneğine sahiptir. Ama gerçek hayatta genellikle durağanlık olmadığından, ağların eğitimlerinin sürekli yenilenmesi gerekmektedir. ART ağları sürekli öğrenme ve beklenmedik durumların üstesinden gelme yetenekleri ile sınırsız karmaşıklık altında çalışabilmektedirler.
- ART ağları kendi kendine öğrenen yapıya sahiptirler. Hiçbir danışmanın yardımına ihtiyaç duymadıkları gibi, yeni karşılaştıkları bir durumu sınıflandıramazlarsa bu yeni durum için yeni bir sınıf oluşturarak ağın büyümesini sağlarlar. Bu sayede sınırlı bir yapıdan kurtulmuş, ağın kapasitesi dahilinde bir sınırsızlık elde edilmiş olur.
- ART ağlarında girdi otomatik olarak normalize edilir ve çok düşük veya çok fazla değerdeki girdilerin etkisi azaltılmış olur.

5. Bölüm: Yapay Sinir Ağı Tasarımı

Bir sinir ağı modeli oluşturmak için nöronların bağlanış şekli (Topoloji), işlemci elemanların kullandıkları toplama ve aktivasyon fonksiyonları, öğrenme metodu, öğrenme kuralı ve algoritması belirlenmelidir. Eldeki veriye ve ağdan yapmasını istediğiniz uygulamanın şekline göre model tasarlanır. Kurulan modelin başarısı modelin mimarisinin doğru oluşturulması ile direk ilgilidir. Bunun için YSA tasarımcısının, ağın yapısına ve işleyişine ilişkin şu kararları vermesi gerekmektedir.

- Ağ mimarisinin seçimi ve yapısal özelliklerinin belirlenmesi (katman sayısı ve katmandaki işlemci eleman sayısı gibi),
- İşlemci elemanların kullandığı fonksiyonların karakteristik özelliklerinin belirlenmesi,
- Öğrenme algoritması ve parametrelerinin belirlenmesi,
- Eğitim ve test setinin oluşturulması.

Bu kararlar doğru verilmediği takdirde sistem karmaşıklığı artacak veya kararlı ve istikrarlı sonuçlar alınamayacaktır. Yapay sinir ağının toplam tepki süresi, eğitim süresi, sistem karmaşıklığı ile artar. Sistem tasarlanırken uygulamanın ne kadar süreceği, hafızada ne kadar yer kaplayacağı gibi bilgiler hesaplanmalıdır.

5.1. Öğrenme Algoritması ve Yapay Sinir Ağı Topolojisinin Seçimi

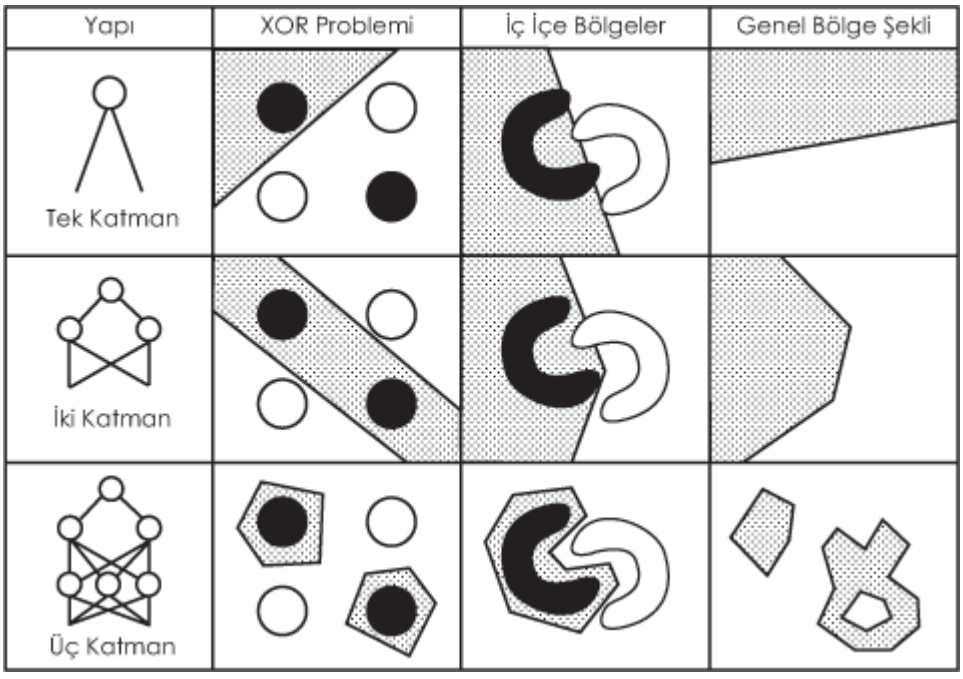
Bir problemin çözümü için uygulanacak olan YSA topolojisi öncelikle problemin türüne bağlı olmaktadır. Hangi ağ modelinin hangi problemin çözümü için daha uygun olduğunun bilinmesi oldukça önemlidir. Bölüm 3.3'de hangi YSA topolojisinin hangi problemlerin çözümünde kullanıldığı özetlenmiştir.

Probleme uygun YSA topolojisinin seçimi ağda kullanılması düşünülen öğrenme algoritmasına bağlıdır. Bir çok YSA yapısı tek bir öğrenme algoritması ile kullanılabildiğinden öğrenme algoritmasının seçimi ile kullanılacak YSA topolojisi de zorunlu olarak seçilmiş olacaktır.

5.2. Katman ve Nöron Sayısının Seçimi

Ağ yapısında kullanılan katman sayısı ve her katmanda bulunan nöron sayısı yapay sinir ağının karmaşıklığını belirler. Katman sayısı ve katmanlardaki nöron sayısı arttıkça yapay sinir ağının işlem ve öğrenme yeteneği artarken yakınsama süresi de artmakta, ağın genelleme kabiliyeti düşmekte ve ağın ezberlemesine (Memorization) neden olmaktadır. Gereğinden az kullanımı ise verilerdeki desenin yeteri kadar öğrenilememesine yol açar. Çoğu problem için iki veya üç katmanlı bir ağ tatmin edici sonuçlar üretebilmektedir.

İşlemci eleman sayısı ve katman sayısını belirlemenin en iyi yolu, birkaç deneme yapılarak ağın performansına bağlı olarak en uygun katman sayısına karar vermektir. Bunun için izlenecek yol, başlangıçtaki nöron sayısını istenilen performansa ulaşıncaya kadar arttırmak veya tersi şekilde istenen performansın altına inmeden azaltmaktır. Şekil 5.1'de katman sayısının iki boyutlu desenlerin öğrenilmesinde yarattığı farklar geometrik olarak gösterilmiştir.



Şekil 5.1 Sınıflandırma işleminde, gizli katman sayısının iki boyutlu örnek uzayındaki rolünün geometrik gösterimi (Jain, Mao ve Mohiuddin, 1996, s. 9).

Ayrıca Bölüm 4.1'de bahsedilen, düzlemde (Hyperplane) düz bir çizgi ile keserek Karar Bölgesi (Decission Region) oluşturan algılayıcının aksine, çok katmanlı ileri beslemeli ağlar, bir gizli katman yardımı ile, düzlemi konveks bölgelere ayırabilme yeteneğine sahiptir. Gizli katman sayesinde karar bölgeleri doğrusallıktan çıkarılmış, farklı türlerde konveks geometrik şekillere kavuşulmuş olunur. En azından her bir karar bölgesi için bir gizli işlemci eleman gerekmektedir. Eğer karar bölgeleri fazla ayrık veya üst üste iseler ikinci bir gizli eleman gerekmektedir. Şekli 5.1'de sınıflandırma işleminde gizli katman sayısının iki boyutlu örnek uzayındaki rolünün geometrik gösterimi örneklendirilmiştir.

5.3. Fonksiyonların Seçimi

İşlemci elemanların karakteristik özelliklerinin belirlenmesi de yapay sinir ağının tasarımında alınacak önemli kararlardan biridir. Toplama fonksiyonunun ve aktivasyon fonksiyonunun seçimi büyük ölçüde verilerin özelliklerine ve ağdan öğrenilmesi istenen verinin türüne ve yapısına bağlı olmaktadır. Aktivasyon fonksiyonları içinde en çok kullanılanlar doğrusal, işaret, eşik, sigmoid, hiperbolik tanjant ve lojistik fonksiyonlarıdır. Doğrusal olmayan problemlerde doğrusal olmayan aktivasyon fonksiyonları kullanılmaktadır.

5.4. Normalizasyon

Veriler ağa sunulmadan önce normalizasyon işlemine tabi tutulurlar. Verideki aşırı salınımları engellemek ve sistem performansını arttırmak için kullanılırlar. Bunun için logaritmik fonksiyonlar kullanıldığı gibi, genellikle verilerin [0,1] veya [-1,+1] aralıklarından birine ölçeklendirilmesi önerilmektedir. Ölçekleme verilerin geçerli eksen sisteminde sıkıştırılması anlamı taşıdığından veri kalitesi aşırı salınımlar içeren problemlerin YSA modellerini olumsuz yönde etkileyebilir (Saraç, 2004, s.35).

5.5. Performans Fonksiyonunun Seçilmesi

Performans fonksiyonları, istenilen çıktı değerleri ile ağın ürettiği değerler arasındaki farkların kümülatif değerleri hesaplamaktadır. Hesaplanan bu değerler sayesinde ağın, eğitim setinin gösterdiği desene ne kadar yaklaştığı gözlenmekte ve

bağlantıların ağırlık değerleri bu bilgiler kullanılarak değiştirilmektedir. Bu nedenle, performans fonksiyonları, öğrenme performansını etkileyen önemli hususlardan biridir. İleri beslemeli ağlarda kullanılan tipik performans fonksiyonu **Hata Kareleri Ortalaması**'dır (Mean Square Error, MSE).

$$MSE = \frac{1}{n}\sum_{i=1}^{n}[e(t)]^2 \qquad \textbf{(5.1)}$$

Ortalama Hata (Mean Error, ME), **Hata Kareleri Ortalamasının Karekökü** (Root Mean Square) ve **Ortalama Mutlak Hata** (Mean Absolute Error, MAE) Kullanılan diğer performans fonksiyonlarından bazılarıdır. Sırası ile aşağıdaki denklemlerle ifade edilir:

$$ME = \frac{1}{n}\sum_{i=1}^{n}e(t) \qquad \textbf{(5.2)}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}[e(t)]^2} \qquad \textbf{(5.3)}$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left|e(t)\right| \qquad \textbf{(5.4)}$$

5.6. Öğrenme Katsayısı ve Momentum'un Seçilmesi

Öğrenme sürecinde ağırlıklardaki değişim **Öğrenme Oranı/Katsayısı** *(λ)* ile orantılıdır. Genellikle bu oran **Salınıma** (Oscillation) yol açmayacak kadar büyük alınmaya çalışılır. Öğrenme oranı, ağırlıkların bir sonraki düzeltmede hangi oranda değiştirileceğini belirlemektedir. Ağ üzerinde önemli bir etkisi bulunan öğrenme oranının değeri uygulanan probleme göre değişmekle birlikte büyük bir değer

seçilmesi salınıma yol açmakta ve ağın herhangi bir minimum değerine ulaşması zorlaşmaktadır. Küçük bir değer seçilmesi ise öğrenme süresinin uzamasına ve ağın yerel çözümlere takılmasına neden olmaktadır.

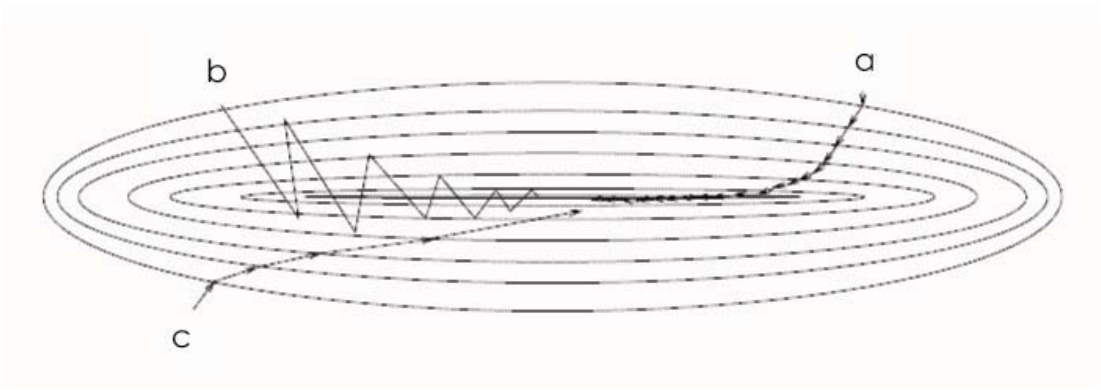
Yerel çözüm şu şekilde açıklanabilmektedir; bir problemin çözümü için en az hatayı veren ağırlık vektörünü pratikte her zaman yakalamak mümkün olmayabilmektedir. Bu çözüm, ağın sahip olabileceği en iyi çözümdür (Mutlak Minimum). Fakat bu çözüme nasıl ulaşılacağı tam olarak bilinmemektedir. Ağ, eğitim sırasında bu çözüme ulaşmaya çalışmaktadır. Ancak bazen ağ farklı bir çözüme takılabilmekte ve performansı daha fazla iyileştirmek mümkün olmamaktadır (Mehrotra, Mohan ve Ranka, 1997).

Karmaşık bir ağda hata yüzeyi fazlasıyla inişli çıkışlıdır. Bu durum, dereceli azaltmadan dolayı ağın yerel minimum tuzağına düşmesiyle sonuçlanır. Olasılık yöntemleri bu tuzaktan kurtulmaya yardımcı olsa da oldukça yavaş kalmaktadır. Bir başka öneri, gizli katman veya gizli katmandaki nöron sayısının arttırılmasıdır. Bu yöntem işe yarasa da hata yüzeyinin boyut sayısını da arttırmaktadır. Gizli nöron sayısının bir üst limiti her zaman olacaktır, çünkü araştırmalar gizli nöron sayısının fazla olduğu durumlarda ağın ezberlediğini ortaya koymuşlardır.

Büyük öğrenme oranlarında salınıma imkan vermemenin bir yolu **Momentum** teriminin eklenmesidir.

Momentumun oynadığı rol Şekil 5.2'de gösterilmiştir. Eğer momentum terimi kullanılmazsa ağın düşük öğrenme katsayısı ile yerel minimuma ulaşması çok uzun sürmekte, büyük öğrenme katsayısında ise salınımdan dolayı yerel minimuma asla ulaşamamaktadır. Momentum kullanıldığında ise yerel minimuma hızlı bir şekilde ulaşılabilmektedir.

Özetle momentum, özellikle yerel çözümlere takılan ağların bir sıçrama ile daha iyi sonuçlar bulmasını sağlamak amacıyla kullanılmaktadır. Bu değerin küçük olması yerel çözümlerden kurtulmayı, büyük bir değer olması ise tek bir çözüme ulaşmayı zorlaştırabilmektedir.



Şekil 5.2 Ağırlık Uzayındaki Düşme: a) düşük öğrenme oranı, b) büyük öğrenme oranı: salınım, c) momentum terimi eklenmiş büyük öğrenme oranı (Gurney, 1996, s.37)

5.7. Performans Faktörleri

Performansın arttırılması şu faktörlere bağlıdır:

- **Öğrenme algoritması ve iterasyon (epoch) sayısı**: Hatanın minimize edilmesi için gereklidir. Örneklerin ağa tekrar tekrar gösterilmesi ile ağ hata oranını minimum yapmaya çalışmaktadır.
- **Eğitim setindeki örnek sayısı**: Örneklerin gerçek fonksiyonu temsil etmesi için gereklidir.
- **Gizli işlemci eleman sayısı**: Ağın gücüdür. Yumuşak dalgalı fonksiyonlarda oldukça az gizli elemana ihtiyaç duyulurken, sert inişli çıkışlı fonksiyonlarda daha fazla gizli eleman kullanılmalıdır.

İlk olarak doğru hata ölçütü tanımlanmalıdır. Tüm öğrenme algoritmaları, eğitim boyunca eğitim setinin hatalarını minimize etmeye çalışır. Eğitim setindeki her bir örnek için ortalama hata **Öğrenme Hata Oranı** olarak ifade edilir ve şu şekilde formülüze edilir:

$$E_{Eğitim} = \frac{1}{P_{Eğitim}} \sum_{p=1}^{P_{Eğitim}} E^p \qquad \textbf{(5.5)}$$

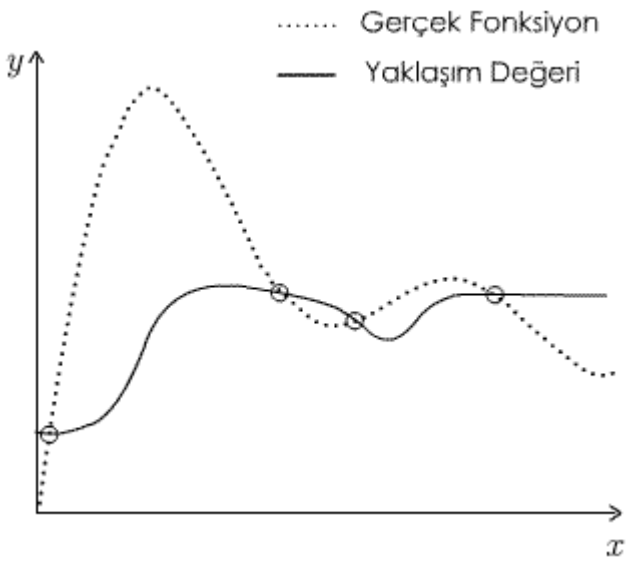
Daha önceki denklemlerde de ifade edildiği gibi, E^p eğitim örneğinin gerçek çıktı değeri ile ağın çıktı değeri arasındaki farktır:

$$E^p = \frac{1}{2}\sum_{o=1}^{N_o}\left(d_o^p - y_o^p\right)^2 \qquad \textbf{(5.6)}$$

Bu hata, eğitim süresince ölçülebilen bir hatadır ve açıkça görüldüğü gibi eğitim setindeki her bir örnek için farklı değerler almaktadır. Bu hata sistemin performansının doğru bir şekilde ölçülmesi açısından tüm örnekler için genişletilmeli ve sistem için tek bir hata ölçütü ortaya konmalıdır. Sonrasında test verilerinin de ortalama hata oranlarının hesaplanması gerekmektedir:

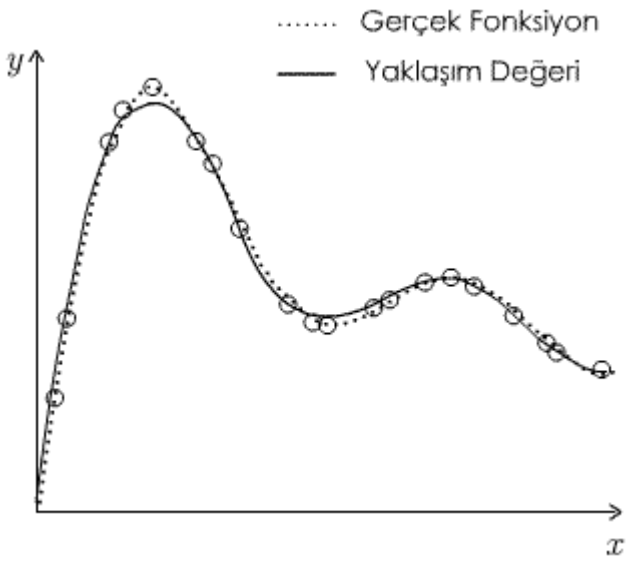
$$E_{Test} = \frac{1}{P_{Test}}\sum_{p=1}^{P_{Test}} E^p \qquad \textbf{(5.7)}$$

Örnek Sayısının Performansa Etkisi: Eğitim setindeki örnek sayısı ne kadar fazla olursa, ağ, fonksiyon üzerindeki bilinen nokta sayısı fazla olacağından, fonksiyon hakkında daha yakın bir kestirim yapma imkanı bulacaktır. Bir *y = f(x)* fonksiyonunun ileri beslemeli bir ağ ile yaklaştırılacağı (Function Approximation) varsayılsın. Ağın, bir girdi elemanına, sigmoid aktivasyon fonksiyonlu 5 gizli elemana ve doğrusal aktivasyon fonksiyonlu bir çıktı birimine sahip olduğu ve az sayıda eğitim örneği ile eğitildiği varsayılsın. Şekil 5.3, 4 örnekle eğitilen böyle bir ağın, eğitim sonuç grafiğini göstermektedir.



Şekil 5.3: Örnek Sayısının Fonksiyon Yaklaştırma Üzerindeki Etkisi (4 Örnek)

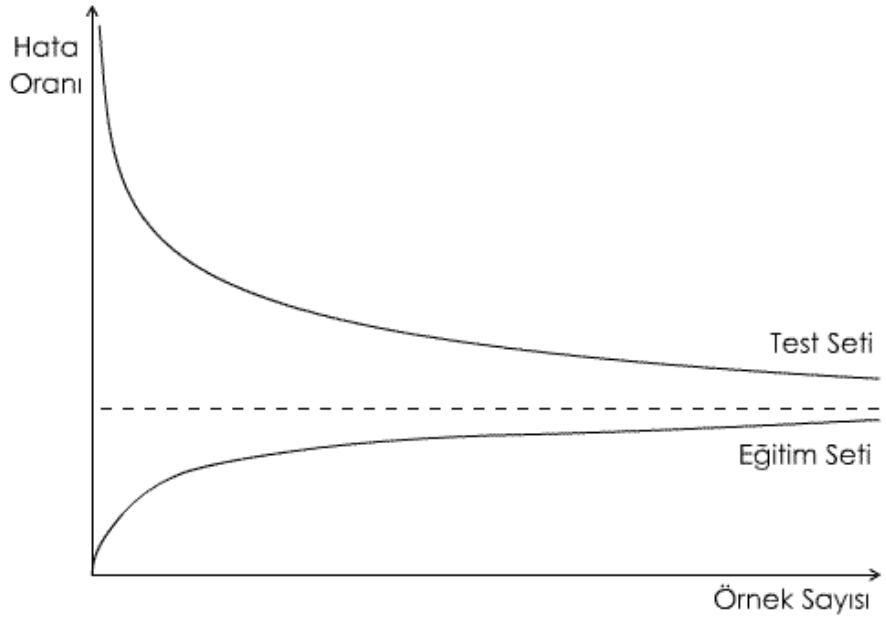
Şekilde gerçek fonksiyon, kesikli çizgi ile gösterilirken, eğitim sonrası yapay sinir ağının yaklaşımı düz çizgi ile gösterilmiştir. İki eğrinin kesim noktaları ağa öğrenme sürecinde verilen örneklerdir. Aynı ağa 4 yerine 20 örnek gösterildiğinde Şekil 5.4'deki grafik ortaya çıkacaktır.



Şekil 5.4: Örnek Sayısının Fonksiyon Yaklaştırma Üzerindeki Etkisi (20 Örnek)

İlk durumda az örnekle $E_{Eğitim}$ oldukça az olacak ama E_{Test} oldukça yüksek çıkacaktır. İkincisinde ise $E_{Eğitim}$ nisbeten daha fazla çıksa da çok düşük E_{Test} değeri ile karşılaşılacaktır. Aynı ağ, farklı örnek sayıları ile eğitildiğinde ve her örnek seti ağa

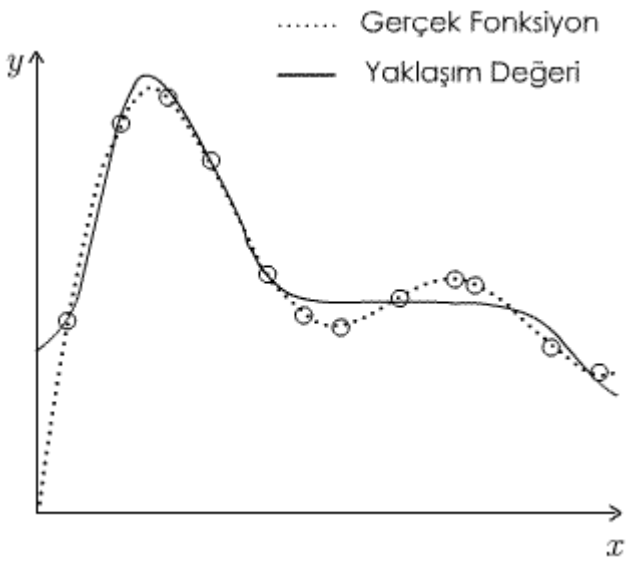
onar defa gösterildiğinde, Şekil 5.5'deki ortalama öğrenme ve test hatası oranı grafiği ile karşılaşılacaktır:



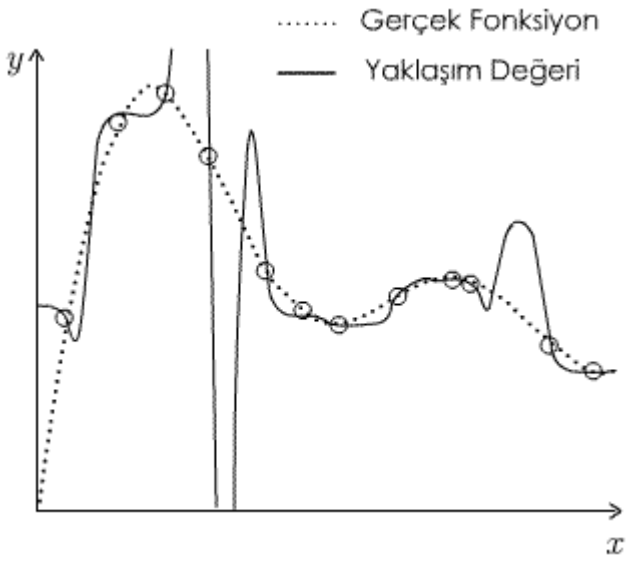
Şekil 5.5: Örnek Sayısı İle Hata Oranları Arasındaki İlişki

Görüldüğü gibi, eğitim hatası, örnek sayısı arttıkça artarken, test hatası düşmektedir. Eğitim hatasının düşük olması, ağın performansının iyi olduğunu her zaman göstermez. Eğitim setinin örnek sayısı arttırıldıkça her iki hata oranı da aynı değere yakınsamaktadır. Bu değer, ağın temsil gücü ile ilişkilidir. Bu değer aynı zamanda gizli eleman sayısına ve kullanılan aktivasyon fonksiyonuna bağlıdır. Eğer bu iki hata oranı aynı değere yakınsamazlarsa, eğitim süreci için mutlak minimum bulunması imkansızlaşır.

Gizli Nöron Sayısının Performansa Etkisi: Aynı fonksiyonun yaklaşımı için, aynı ağın kullanıldığını, ağın bu kez farklı gizli işlemci eleman sayılarıyla eğitildiği varsayılsın. Şekil 5.6, ağın gizli katmanında 5 nörona, Şekil 5.7 ise 20 gizli nörona sahip olması durumunda, eğitim sonrasındaki davranışını göstermektedir.



Şekil 5.6: Gizli Nöron Sayısının Fonksiyon Yaklaştırma Üzerindeki Etkisi (5 Gizli Nöron)

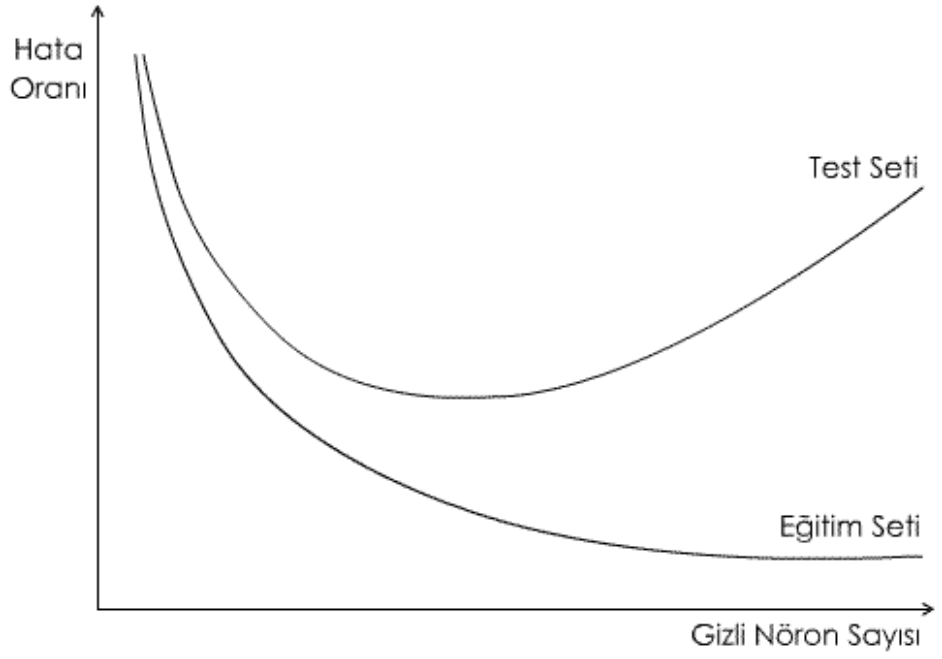


Şekil 5.7: Gizli Nöron Sayısının Fonksiyon Yaklaştırma Üzerindeki Etkisi (20 Gizli Nöron)

Şekil 5.6'daki etki, **Fazla Eğitim** (Overtraining veya Overfitting) olarak adlandırılmaktadır. Ağ, eğitim verilerini körlemesine öğrenmiş ama test verilerinde aynı performansı gösterememiştir, bu nedenle genelleme yeteneği zayıftır. Şekil 5.7'de ağ, örneklere tam olarak uymaktadır fakat fazla gizli eleman

kullanıldığından, yaklaşım fonksiyonu, gerçek fonksiyondan çok daha sert iniş çıkışlara sahip olmaktadır. Gerçek hayatta, eğitim örnekleri **Gürültü** (Noise) içermekte ve örnekte kullanılan fonksiyon gibi düzgün bir grafik çizmemektedir. Bu nedenle gizli katman sayısının fazlalığı gerçek dünyanın verileri için daha gerçekçi, daha doğru yaklaşımlar sağlayabilmektedir.

Bu örnekte de görüldüğü gibi gizli eleman sayısının fazlalığı eğitim hata oranının düşmesini sağlarken, test hata oranını arttıracaktır. Fazladan gizli eleman eklemek $E_{Eğitim}$'i her zaman düşürecektir. Fakat, E_{Test} ilk başlarda azalacak, sonrasında artacaktır. Aynı ağ, aynı eğitim seti fakat farklı gizli işlemci eleman ile test edildiğinde ortalama eğitim ve test hatası oranlarının grafiği Şekil 5.8'deki gibi olmaktadır.



Şekil 5.8: Gizli Nöron Sayısı ile Hata Oranları Arasındaki İlişki

6. Bölüm: Uygulama

Bu bölümde, çalışmada kullanılan modeller tanıtılmakta ve çalışma kapsamında, Türkiye ekonomisine ait bir değişkenin tahminine yönelik YSA modeli kurulmuştur. Bunun yanında, YSA yönteminin tahmin gücünü karşılaştırabilmek amacıyla ekonometrik yöntem olan çoklu doğrusal regresyon analizi kullanılarak aynı değişkenler modellenmiştir. Karşılaştırılabilir sonuçlar elde edebilmek amacıyla mümkün olduğunca benzer yapıda modeller kullanılmaya çalışılmıştır. Her iki modellemenin sonuçları gösterilip yorumlanmış, çalışmanın sonuç kısmında her iki modelin karşılaştırılması yapılmıştır.

6.1. Veri Setinin Tanıtılması

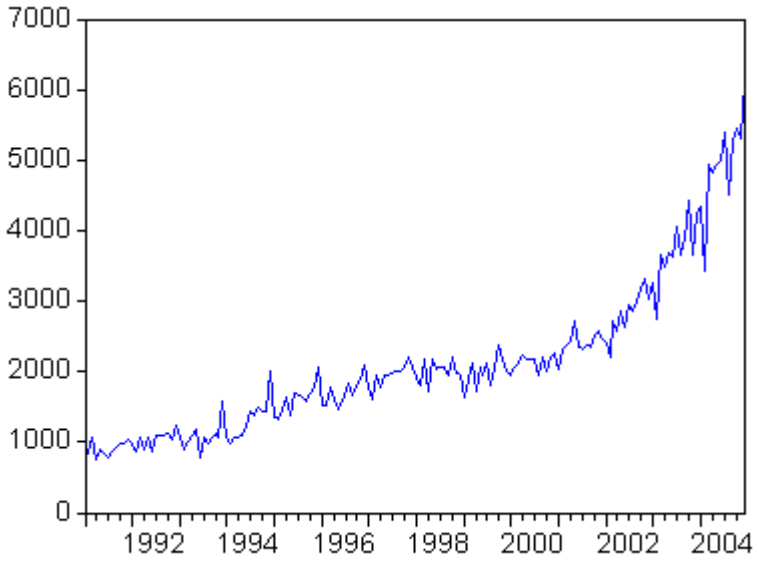
Uygulamada, imalat sanayi aylık ihracat değerleri (IHRACAT) üzerinde etkisi olduğu düşünülen bağımsız (açıklayıcı) değişkenler; ABD Doları (döviz alış) aylık ortalama değeri (DOLAR), 1997=100 bazlı aylık toplam sanayi sektörü sanayi üretim endeksi (SUI), ulusal sınai endeksi kapanış fiyatları (USE), 1997=100 bazlı aylık imalat sanayi kısmi verimlilik endeksi (üretimde çalışılan saat başına, VERSAAT) olarak belirlenmiştir. Belirtilmesi gereken diğer bir nokta ise modelleme yapısı olarak geriye dönük modellemenin kullanılmış olmasıdır. Diğer bir ifadeyle, modeller açıklayıcı değişkenlerin gecikmelerini içermektedir ve bunlara altıncı açıklayıcı değişken olarak bağımlı değişkenin gecikmesi eklenmiştir.

Ekonometrik modelde bazı değişkenler birbirleri ile ilişkili olduklarından çoklu doğrusal bağlantı problemi ile karşılaşılmıştır. Aynı zamanda, çalışılan değişkenler zaman serisi özelliği taşıdığından çoklu doğrusal regresyon modelinde ele alınan değişkenlerin trend etkisi ekonometrik modelde otokorelasyon sorununa, yığılım etkisi ise değişen varyanslılığa (Heteroskedasticity) yol açmış, uygun dönüşümler yapılarak model tahmine uygun hale getirilmiştir.

Söz konusu değişkenlerle dönüşüm yapılmadan elde edilen model, çoklu doğrusal regresyon analizi varsayımlarını sağlamadığından değişen varyanslılık sorunu için

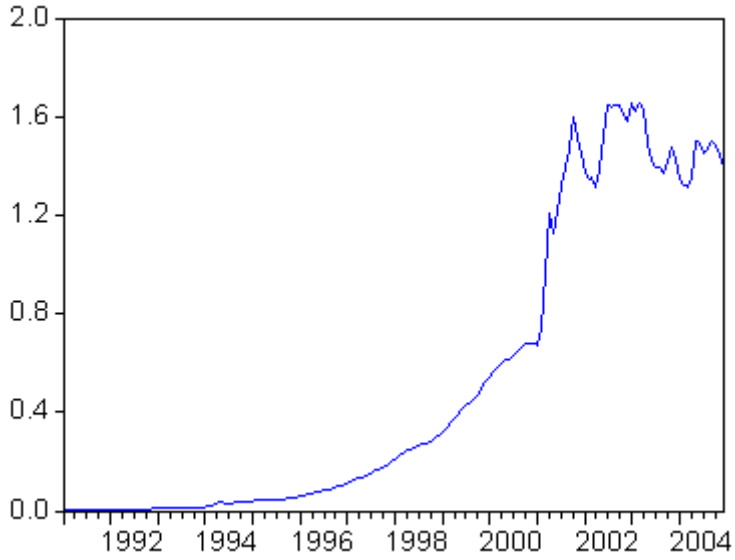
doğal logaritmik dönüşüm uygulanmıştır. Bu dönüşüme ek olarak, çoklu doğrusal bağlantının giderilmesi için, bağlantıya yol açan USE ve VERSAAT değişkenlerinin elenmesi yoluna gidilmiştir. Regresyon modelinde kullanılacak değişken seçimi konusunda birçok alternatif model denenmesi sonucunda, otoregresif model yapısının varsayımları sağladığı ve istatistiki olarak anlamlı olduğu belirlenmiştir. Bu model sayesinde otokorelasyon probleminin de ortadan kaldırıldığı gözlenmiştir. Çoklu doğrusal regresyon tahmin modeli belirlendikten sonra, YSA modeli aynı değişkenler için uygulanarak tahmin karşılaştırması yapılmıştır.

Veriler, Türkiye Cumhuriyet Merkez Bankası internet sayfasından alınmıştır. Çalışmada değişkenlerin, 1991:01 ile 2004:12 arasındaki döneme ait verilerinin analizi yapılmıştır. Söz konusu döneme ilişkin model değişkenlerinin grafikleri incelenmiş ve yorumları yapılmıştır.



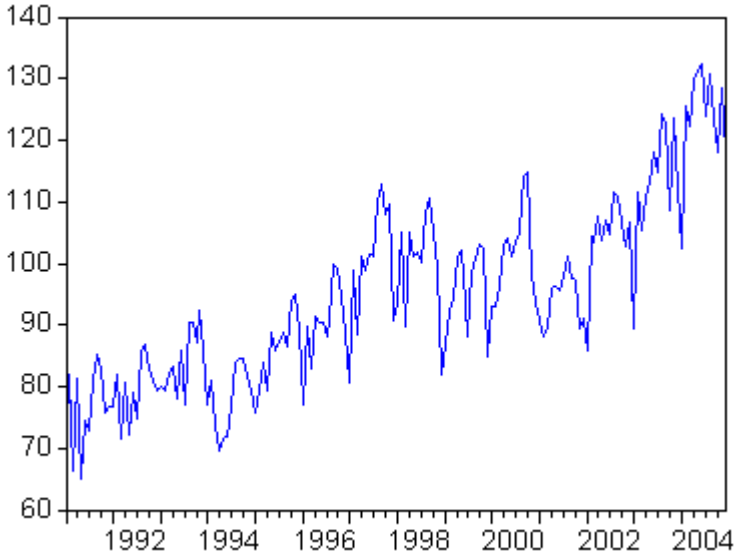
Şekil 6.1 İmalat Sanayi İhracat Değerleri (Aylık, Milyon $)

Şekil 6.1 'deki grafikte, Ocak 1991 ile Aralık 2005 tarihleri arasında IHRACAT serisinde yığılım etkisi görülmektedir. İlk yıllardaki artan trend etkisi son yıllarda yerini üstel büyümeye bırakmıştır.



Şekil 6.2 ABD Doları (Döviz Alış) Değerleri (Aylık, Ortalama, YTL)

Şekil 6.2'deki grafikte, analizi yapılan dönemde, DOLAR serisinde ilk yıllarda artan trend etkisi ve 2001 yılında yapısal bir kırılmanın yanı sıra bu dönem sonrası yığılım etkisi görülmektedir.



Şekil 6.3 Toplam Sanayi Sektörü Sanayi Üretim Endeksi (Aylık, 1997=100)

Şekil 6.3'teki grafikte, SUI serisinde hem artan bir trend etkisi hem de yığılma etkisi görülmektedir.

6.2. Çoklu Doğrusal Regresyon Modeli

Çoklu doğrusal regresyon modelinde daha önce de bahsedildiği gibi değişkenler doğal logaritmik dönüşüme uğratılarak bir gecikmeli değerleri kullanılmıştır. Çoklu doğrusal regresyon için genel fonksiyonel gösterim Denklem (6.1)'deki gibidir.

$$log(ihracat) = f(log(ihracat_{-1}), log(dolar_{-1}), log(sui_{-1})) \qquad \textbf{(6.1)}$$

Modelin belirlenmesinde bazı teorik kısıtlar göz ardı edilmiştir. Bunun nedeni, çalışmanın bir politika analizi olmayıp, iki yöntem arasındaki performans karşılaştırması amacını taşımasıdır. Bu sebeple bağımsız değişkenlerin belirlenmesinde, varsayımları sağlayan ve istatistiksel anlamlılığı veren az sayıda etkin değişken ele alınmıştır. Çünkü, yapay sinir ağları modelleri ile sınırsız sayıda bağımsız değişken kullanılarak analiz yapılabilmesine karşın, regresyon analizinde sınırlı sayıda ve kendi içinde çoklu doğrusal bağlantı bulunmayan bağımsız değişkenler kullanılmak gerekmektedir. Bu yüzden, her iki yöntem için en uygun modelleri bulmak yerine belirli bir modelle çalışılarak, karşılaştırma yapılması sağlanmıştır. Çoklu doğrusal regresyon modeli için Quantitive Micro Software Eviews 4.1 paket programı kullanılmış ve elde sonuçlar Tablo 6.1'de gösterilmiştir.

Tablo 6.1 Çoklu Doğrusal Regresyon Modeli İstatistik Değerleri

Dependent Variable: LOG(IHRACAT)
Method: Least Squares
Date: 06/24/06 Time: 11:42
Sample(adjusted): 1991:02 2004:12
Included observations: 167 after adjusting endpoints

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	-0.489026	0.462804	-1.056659	0.2922
LOG(IHRACAT(-1))	0.579993	0.051134	11.34254	0.0000
LOG(DOLAR(-1))	0.041213	0.010547	3.907614	0.0001
LOG(SUI(-1))	0.825123	0.104381	7.904920	0.0000

R-squared	0.948392	Mean dependent var	7.542452
Adjusted R-squared	0.947442	S.D. dependent var	0.477865
S.E. of regression	0.109553	Akaike info criterion	-1.561150
Sum squared resid	1.956315	Schwarz criterion	-1.486468
Log likelihood	134.3560	F-statistic	998.4686
Durbin-Watson stat	2.092506	Prob(F-statistic)	0.000000

Modelin bağımsız değişkenlerinin her biri için $p<0.01$ olduğundan modelde yer alan bağımsız değişkenlerin katsayılarının sıfırdan farklı olduğunu iddia eden H_1 hipotezi

%1 anlamlılık düzeyinde kabul edilerek, modelin istatistiksel olarak anlamlı olduğu sonucuna ulaşılmıştır. Modelin değişkenlerine ait katsayıların işaretleri iktisat teorisine ters düşmemektedir. Modelin açıklayıcılığı %94,8 olarak elde edilmiştir. Modelin anlamlılığını belirten H_1 hipotezi F testi sonucunda, $p<0.01$ olduğundan %1 anlamlılık düzeyinde kabul edilmiştir. Daha sonra varsayımlar test edilerek modelin tahmine uygunluğu araştırılmıştır. İlk olarak otokorelasyon sınamasında Breusch-Godfrey Seri Korelasyon LM testi uygulanmıştır. Testin sonuçları Tablo 6.2'de görülmektedir.

Tablo 6.2 Breusch-Godfrey Seri Korelasyon LM Testi

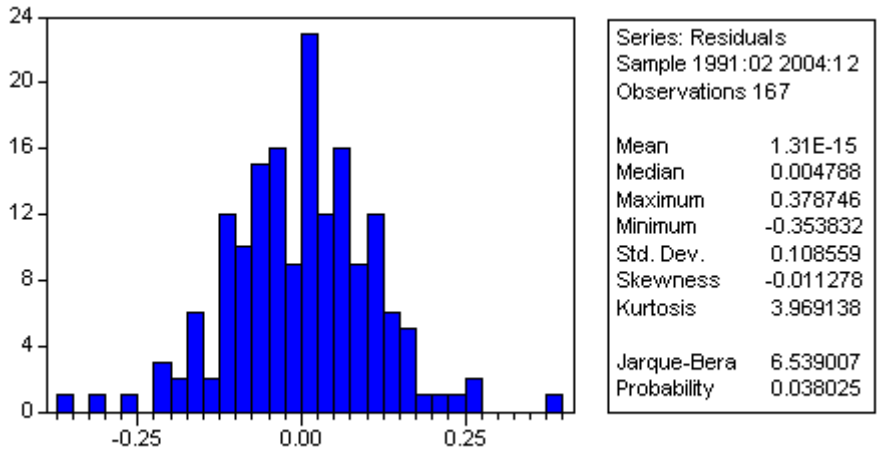
F-statistic	1.129919	Probability	0.289375
Obs*R-squared	1.156725	Probability	0.282146

LM testi sonucunda $p>0.01$ olduğundan tahmini modelin hata serisinde otokorelasyon olmadığını iddia eden H_0 hipotezi kabul edilmiştir. Değişen varyanslılık testi için White Heteroskedasticity testi kullanılmıştır. Testin sonuçları Tablo 6.3'te görülmektedir.

Tablo 6.3 White Heteroskedasticity Testi

F-statistic	1.591781	Probability	0.121924
Obs*R-squared	13.96429	Probability	0.123601

White testi sonucunda $p>0.01$ olduğundan modelde değişen varyans sorunu olmadığını iddia eden H_0 hipotezi kabul edilmiştir. Ayrıca, model hatalarının normallik varsayımının sınanması için Jarque-Bera testi kullanılmıştır. Testin sonuçları Şekil 6.4'te görülmektedir.



Şekil 6.4 Hatalar Jarque-Bera Testi

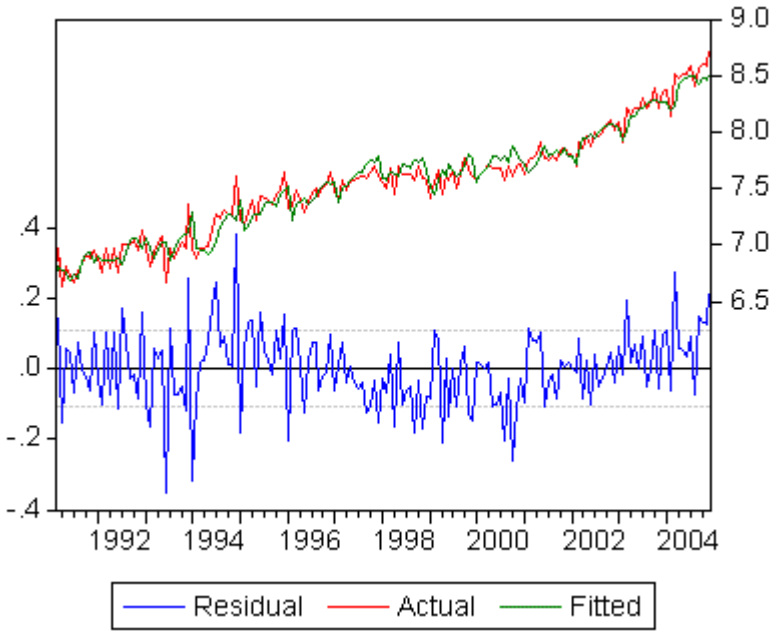
Jarque-Bera test istatistiği p>0.01 olduğundan hataların normal dağıldığını iddia eden H_0 hipotezi kabul edilir. Kurulan modelde çoklu doğrusal bağlantının olup olmadığının araştırılması için VIF (Variance Inflation Factors) değerlerine bakılmıştır. Değerler Tablo 6.4'te gösterilmiştir.

Tablo 6.4 Bağımsız Değişkenler, İstatistik Değerleri (SPSS Programı)

Model		Unstandardized Coefficients		Standardized Coefficients			Collinearity Statistics	
		B	Std. Error	Beta	t	Sig.	Tolerance	VIF
1	(Constant)	-0.489	0.462		-1.056	0.336		
	LOG(DOLAR(-1))	0.041	0.010	0.177	3.907	0.000	0.167	6.005
	LOG(SUI(-1))	0.825	0.104	0.262	7.904	0.000	0.282	3.544
	LOG(IHRAC(-1))	0.579	0.051	0.577	1.,342	0.000	0.131	7.648

Uygulamada 10'un üzerindeki VIF değerlerinin ciddi doğrusal bağlantı göstergesi olduğu kabul edilmektedir (Orhunbilge, 2002, s.242). Tablo 6.4'ten VIF değerleri 10'dan küçük olduğundan çoklu doğrusal bağlantının olmadığı görülmektedir.

Modelin hata serisi ve tahmin değerleri ile gerçek değerlerine ilişkin grafik Şekil 6.4'te gösterilmiştir.



Şekil 6.5 Gerçek, Tahmin Değerleri ve Hata Grafiği

Grafikte, gerçekleşen ve tahmin edilen IHRACAT serilerinin birbiri ile örtüştükleri ve aralarındaki sapmaların aşırılık göstermediği görülmüştür. Modelin test istatistiklerinden de görüldüğü gibi modelin tahmine uygun olduğu açıktır.

6.3. Yapay Sinir Ağı Modeli

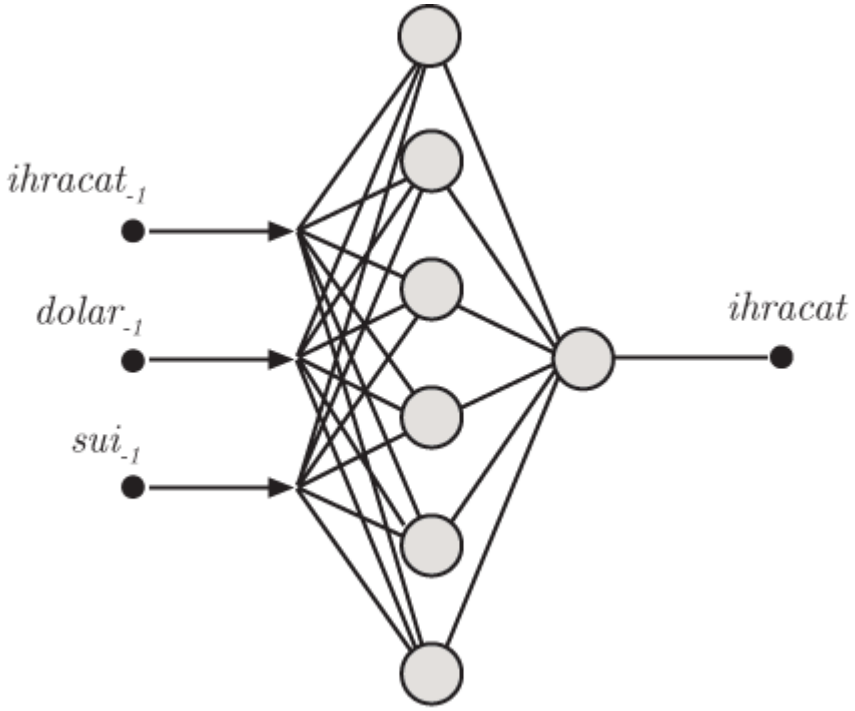
Uygulamada kullanılan YSA modeli, Bölüm 4.2'de anlatılan ileri beslemeli geri yayılım (Back-Propagation) ağıdır. Bu ağın tercih edilmesinin sebebi ekonometrik verilerin modellenmesi ve tahmin çalışmalarında en çok kullanılan model olması ve hem doğrusal hem de doğrusal olmayan modellerdeki tahmin başarısıdır. Ayrıca kullanım kolaylığı ve yakınsama hızı da bu ağ modelinin seçilmesindeki etkenlerden biridir. Model Alyuda Research INC. Alyuda NeuroIntelligence 2.2.577 paket programı kullanılarak kurulmuştur.

Daha önce de belirtildiği gibi, veri setleri ağa sunulmadan önce bir normalizasyon işleminden geçirilmelidirler. Bu amaçla, model kurulumundan önce verilerin analizi program tarafından otomatik olarak yapılmış, veri seti [-1,1] aralığına indirgenerek normalizasyon işlemi yapılmıştır.

Ardından verilerin eğitim, doğrulama ve test işlemleri için bölünmesi işlemi yapılmıştır. 166 adet verinin rasgele seçilen 25 satırı test, yine rasgele seçilen 25 satırı da doğrulama seti (Validation Set) olarak ayrılmış geri kalan 116 satır eğitim seti (Training Set) olarak bırakılmıştır. Son satır program tarafından her üç kümeden de çıkarılmıştır.

Modelde, bağımlı değişken olarak regresyon modelinde olduğu gibi imalat sanayi aylık ihracat değerleri (IHRACAT), bağımsız değişkenler olarak ABD doları (döviz alış) aylık ortalama değeri (DOLAR), 1997=100 bazlı aylık toplam sanayi sektörü sanayi üretim endeksi (SUI) değerleri alınmıştır. Regresyon modeli ile karşılaştırma yapılacağından modellerin bağımsız bir analizinin yapılabilmesi için değişkenler yine regresyon modelinde olduğu gibi doğal logaritmik dönüşüme uğratılmıştır, DOLAR ve SUI değişkenlerinin bir gecikmeli değerleri alınmış ve üçüncü bir bağımsız değişken olarak IHRACAT değişkeninin bir gecikmeli değerleri modele eklenmiştir. Dolayısıyla model üç girdi değişkenine ve bir çıktı değişkenine sahiptir.

Bu üç girdi ve bir çıktı değişkenine sahip model tek gizli katmanlı ileri beslemeli bir geri yayılım ağıdır. Kurulan ağ modelinin mimarisi Şekil 6.6'da gösterilmiştir. Girdi katmanında, ağa sunulan üç bağımsız değişkene ait üç adet girdi işlemci elemanı, çıktı katmanında ise bağımsız değişkene ait bir adet çıktı işlemci elemanı bulunmaktadır. Ara katmandaki işlemci eleman sayısı ise altı adet belirlenmiştir. Bu sayının belirlenmesi için çeşitli işlemci eleman sayıları ile denemeler yapılmış ve bu sayıda karar kılınmıştır.



Şekil 6.6 Modelin YSA mimarisi

Modelde işlemci elemanların birleştirme fonksiyonu toplama fonksiyonu olarak belirlenmiş, gizli katmandaki ve çıktı katmanındaki transfer fonksiyonları da kullanılan programın sınırları dahilinde hiperbolik tanjant olarak seçilmiştir. Performans Fonksiyonu olarak da Ortalama Hata Kareleri (Mean Squared Errors, MSE) kullanılmıştır.

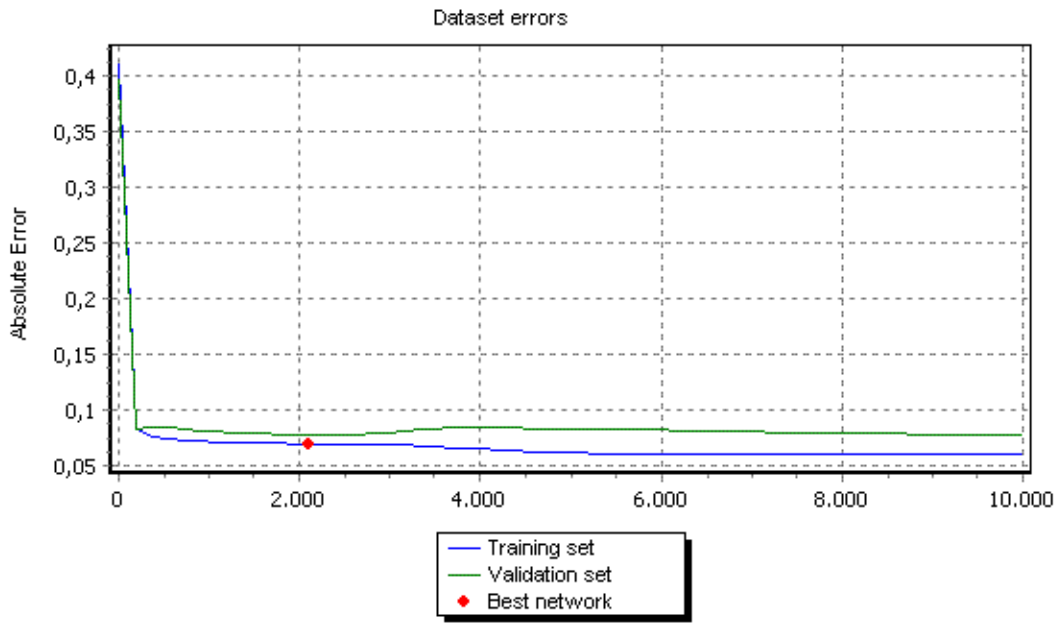
Modelin oluşturulmasının ardından eğitim sürecine geçilmiştir. Ağın eğitimi için çevrimiçi geri yayılım (Online Back Propagation) algoritması kullanılmış, başlangıç öğrenim oranı ve momentum değerleri 0,1 olarak alınmıştır. Bu değerlerin seçimi için farklı başlangıç değerleri ile denemeler yapılmış ve bu değerlerde karar kılınmıştır. Program her iterasyonda öğrenim oranı ve momentum değerlerini değiştirmek üzere ayarlanmıştır. Ağın başlangıç ağırlık değerleri yine program tarafından rastsal olarak atanmıştır.

Model 10.000 döngü (İterasyon, Epoch) kullanılarak eğitilmiş ve bu eğitime ilişkin sonuçlar Tablo 6.5'te gösterilmiştir.

Tablo 6.5 YSA Modeli Hata Sonuçları

	Training	Validation
Absolute Error	0.06021	0.076991
Network Error	0.006295	0
İteration	10001	
Architecture	[3-6-1]	
Training Algorithm	Online Back Propagation	

Eğitim, eldeki veri seti için MSE fonksiyonunu minimize eden ağırlık değerlerinin bulunmasıdır. Bu aşamada eğitim seti ile birlikte doğrulama seti de ağa sunulmakta ve en küçük hata düzeyine ulaşılmaya çalışılmaktadır. Eğitim setinin mutlak hatasının minimum olması her zaman eğitimin doğru olduğu anlamına gelmemektedir. Bölüm 5.2'de de bahsedildiği gibi, öğrenmeden çok, ezberleme noktasına gelinmiş olabilir. Bu durumdan kaçınmak için doğrulama seti kullanılmakta ve en iyi performansı veren ağırlık değerlerinin seçiminde, eğitim sonunda elde edilen doğrulama seti hata değerleri kullanılmaktadır. Şekil 6.7, eğitim seti ve doğrulama setinin 10.000 iterasyon boyunca aldıkları ortalama hata değerleri grafiğini göstermektedir. En düşük hata değeri 2110'uncu iterasyonda yakalanmıştır. Eğitim sonunda ağın bu iterasyondaki ağırlık değerleri program tarafından en iyi değerler olarak atanmıştır.



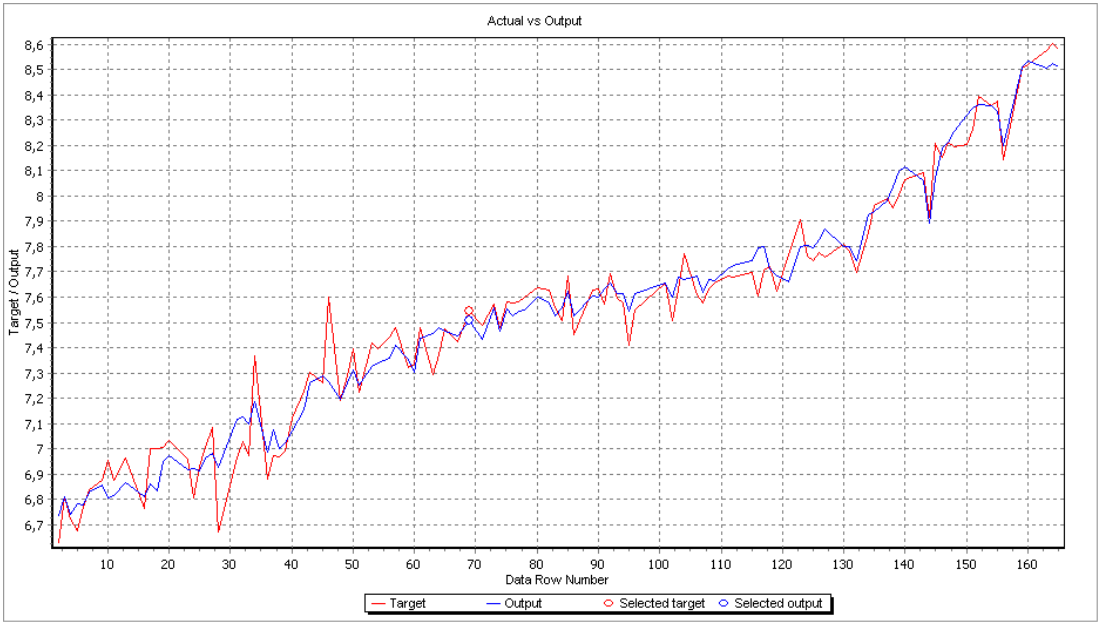
Şekil 6.7 Eğitim Seti İterasyon Hata Grafiği

Bu iterasyona ait ortalama tahmin (Output) ve gerçek değerleri (Target) ile hata değerleri Tablo 6.6'da gösterilmiştir.

Tablo 6.6 En iyi İterasyon Çıktı Değerleri

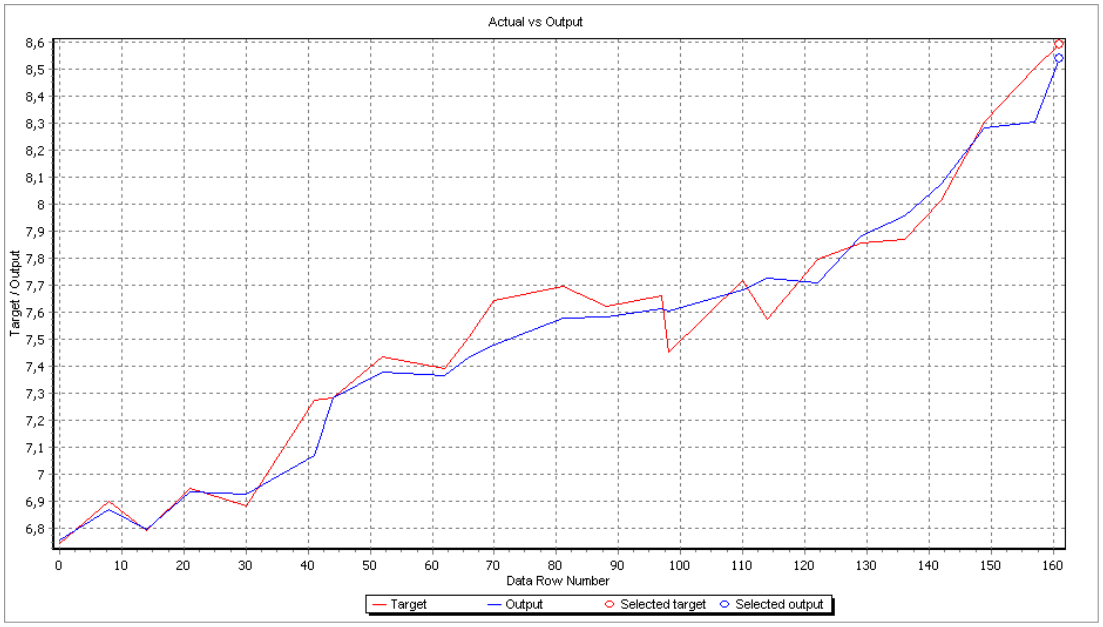
	Target	Output	AE	ARE
Mean:	7.519644	7.525264	0.068813	0.009212
Std Dev:	0.457453	0.465143	0.055091	0.007553
Min:	6.629114	6.702264	0.000599	0.000082
Max:	8.591011	8.564216	0.259259	0.037467
Correlation:	0.981955			
R-Squared:	0.964087			

Tablo 6.6'dan görüldüğü gibi modelin açıklayıcılığı %98,1 olarak elde edilmiştir. En iyi iterasyon sonucunda ağın tahmin değerleri ile gerçek değerlerinin karşılaştırıldığı grafik eğitim seti için Şekil 6.8'deki gibidir.



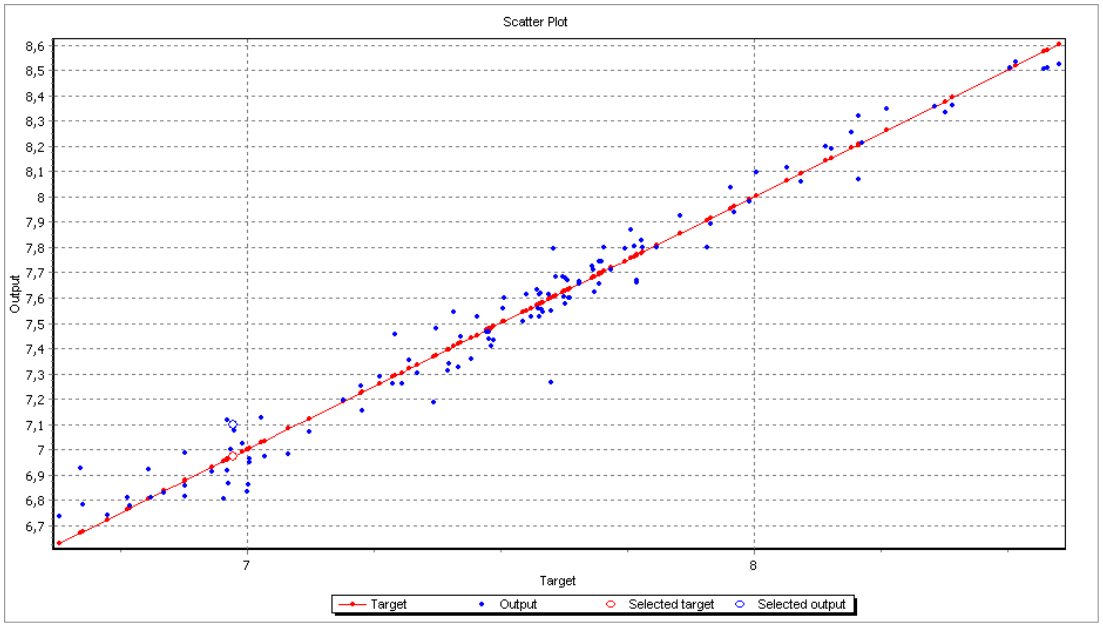
Şekil 6.8 Eğitim Seti Gerçek Değerler ve Tahmin Değerleri Karşılaştırması

Test setine ait gerçek değerler ile tahmin değerlerinin karşılaştırmalı grafiği Şekil 6.9'da verilmiştir.



Şekil 6.9 Test Seti Gerçek Değerler ve Tahmin Değerleri Karşılaştırması

Ağın hem öğrenme, hem de daha önce görmediği test setinde başarılı olduğu Şekil 6.8 ve 6.9'da görülmektedir. Bu sonuç Şekil 6.10'daki gerçek değerler ile tahmin değerleri arasındaki grafikte de görülmektedir.



Şekil 6.10 Gerçek Değerler ve Tahmin Değerleri

Analizin sonucunda elde edilen ağın girdilere ait önem yüzdeleri Tablo 6.7'de gösterilmiştir. Regresyon modeli ile aralarındaki farklar açıkça görülmektedir. Bunun sebebi, YSA'ların girdi değişkenlerinin sonuca katkısını açıklamakta başarılı olmamasıdır.

Tablo 6.7 Girdilere Ait Önem Yüzdeleri

Input column name	Importance, %
İHRACAT	23.3004
DOLAR	38.9381
SUİ	37.7615

Doğrulama ve test veri setlerinin sonuçlarından ağın ezberlemediği görülmektedir. Bu sonuçlar doğrultusunda ağın eğitiminin başarılı olduğu ve tahmin yeteneğinin yüksek olduğu sonucuna varılmıştır.

6.4. Model Karşılaştırması

Kurulan modellerin karşılaştırılması için bazı tahmin değerlendirme analizleri yapılmıştır. Tahmin doğruluğunun ölçülmesinde öncelikli konu bir kayıp fonksiyonunun belirlenmesidir. Kayıp fonksiyonunun belirlenmesi, aynı zamanda uygun bir doğruluk ölçüsünün belirlenmesi anlamına geldiğinden önemlidir. Kullanılacak doğruluk ölçüsünün seçimi için yapılan analizin konusu ve amacı önem kazanmaktadır. Tahmin doğruluğunu ölçmek için kullanılabilecek birçok ölçüt bulunmaktadır. Tahmini değerler ile gerçekleşen değerler arasındaki sapma genellikle $e(t) = x(t) - f(t)$ şeklindeki basit hata terimi baz alınarak hesaplanır. Bu doğruluk ölçütlerinin temelini, sapmayı ölçmek için kullanılan Ortalama Hata (Mean Error, ME) kavramı oluşturur ve

$$ME = \frac{1}{n}\sum e(t) \qquad \textbf{(6.2)}$$

denklemi ile ifade edilir. Ortalama Hata kavramını temel alan ve yaygın olarak kullanılan diğer ölçütler arasında Ortalama Hata Kareleri (Mean Squared Error, MSE), Ortalama Hata Kareleri Kökü (Root Mean Squared Error, RMSE) ve

Ortalama Mutlak Hata (Mean Absolute Error, MAE) sayılmaktadır. Bu ölçütlerin formülleri aşağıda verilmiştir.

$$MSE = \frac{1}{n}\sum [e(t)]^2 \quad \textbf{(6.3)}$$

$$RMSE = \sqrt{\frac{1}{n}\sum [e(t)]^2} \quad \textbf{(6.4)}$$

$$MAE = \frac{1}{n}\sum |e(t)| \quad \textbf{(6.5)}$$

Ayrıca, doğruluk ölçütlerinde tahmin hatası (*e*) yerine tahmin hatasının yüzdesi olan $p(t) = e(t) / x(t)$ oranı da kullanılabilmektedir. Bu durumda ölçütler Ortalama Yüzde Hata (Mean Percent Error, MPE), Ortalama Yüzde Hata Kareleri (Mean Squared Percent Error, MSPE), Ortalama Yüzde Hata Kareleri Kökü (Root Mean Squared Percent Error, RMSPE) ve Ortalama Mutlak Yüzde Hata (Mean Absolute Percent Error, MAPE) olarak adlandırılmaktadır ve formülleri aşağıdaki gibidir.

$$MPE = \frac{1}{n}\sum p(t) \quad \textbf{(6.6)}$$

$$MSPE = \frac{1}{n}\sum [p(t)]^2 \quad \textbf{(6.7)}$$

$$RMSPE = \sqrt{\frac{1}{n}\sum [p(t)]^2} \quad \textbf{(6.8)}$$

$$MAPE = \frac{1}{n}\sum |p(t)| \quad \textbf{(6.9)}$$

Bu çalışmada tahmin edilen çoklu doğrusal regresyon modeli ve yapay sinir ağları modeline ilişkin tahmin doğruluk ölçütleri değerleri Tablo 6.8 ve Tablo 6.9'da görülmektedir.

Tablo 6.8 Tahmin Hatasının Doğruluk Ölçütleri

	ME	MSE	RMSE	MAE
Çoklu Regresyon Modeli	0.00000	0.01171	0.10823	0.08404
Yapay Sinir Ağı Modeli	-0.00562	0.00777	0.08814	0.06881

Tablo 6.9 Tahmin Hatasının Yüzde Doğruluk Ölçütleri

	MPE	MSPE	RMSPE	MAPE
Çoklu Regresyon Modeli	-0.00020	0.00021	0.01456	0.01123
Yapay Sinir Ağı Modeli	-0.00076	0.00014	0.01191	0.00921

Her iki tabloda da görüldüğü gibi, yapay sinir ağı modeli ortalama hata ve ortalama yüzde hata değerleri dışında çoklu doğrusal regresyon modeline göre daha iyi sonuçlar vermiştir. Bu da kurulan iki modelden yapay sinir ağı modelinin çoklu doğrusal regresyon modeline göre daha başarılı olduğunu göstermektedir. YSA modelinde veri setindeki bazı satırların doğrulama ve test işlemleri için ayrı tutulduğu ve eğitime katılmadığı da düşünülürse yapay sinir ağlarının daha az veri ile daha doğru sonuçlar verdiği görülebilir.

Belirtilmesi gereken bir nokta, yapay sinir ağları yönteminde verilerin regresyon modelindeki gibi herhangi bir dönüşüme ihtiyaç duymadığıdır. Bu çalışmada karşılaştırma yapılan çoklu doğrusal regresyon modeliyle aynı yapıda bir model kurulması gerektiği için verilerin doğal logaritmaları kullanılmıştır. Ancak verilere herhangi bir dönüşüm uygulanmadan yapılan bir denemede de, yapay sinir ağları modelinin her iki modelden de daha iyi sonuçlar verdiği görülmüştür, elde edilen sonuçlar EK 1'de verilmiştir. Bunun nedeni yapay sinir ağlarının doğrusal olmayan yapısıdır.

Bir başka nokta ise yapay sinir ağları modelinin kurulum aşamasında regresyondaki gibi sınırlayıcı kriterlerin bulunmamasıdır. Çoklu doğrusal regresyon analizinde sağlanması gereken beş varsayım en iyi modelin araştırılması sürecini güçleştirilmektedir. Çoklu doğrusal bağlantının önüne geçilmesi için bazı değişkelerin modelden çıkarılması gerekliliği buna bir örnektir. Bu noktadan

bakıldığında yapay sinir ağları modeli kurulumu ve kullanımı açısından çoklu doğrusal regresyon modelinden çok daha kolay ve anlaşılırdır. Değişken elenmeden ileri beslemeli geri yayılım ağı kullanılarak bir başka model oluşturulmuş ve her iki modelden de daha iyi sonuçlar elde edilmiş, sonuçlar EK 2'de verilmiştir.

6.5. Tahmin

Çalışmanın bu kısmında, Bölüm 6.4'te yapılan model karşılaştırmasından bir adım ileri gidilerek, kurulan modellerin gerçekleşen değerler ile kontrolü yapılmıştır. Modeller, ele alınan değişkenlerin 1991:01 ile 2004:12 tarihleri arasındaki değerleri baz alınarak kurulmuştur. Model kurulumundaki amaç daha sonra gerçekleşmesi beklenen değerlerin tahmin edilmesi olduğundan, modele dahil edilen değişkenlerin 2005:01 ile 2005:06 tarihleri arasındaki gerçekleşen değerleri baz alınarak imalat sanayi ihracat değerlerinin tahmin yoluna gidilmiştir.

Çoklu doğrusal regresyon modelindeki değişken katsayıları ve tahmin fonksiyonu Denklem (6.10)'daki gibidir.

$$LOG(IHRAC) = C_0 + C_1.LOG(IHRAC_{-1}) + C_2.LOG(DOLAR_{-1}) + C_3.LOG(SUI_{-1}) \quad \textbf{(6.10)}$$

Çoklu doğrusal regresyon modeli bağımsız değişkenlere ait katsayılar tablo 6.10'da belirtilmiştir.

Tablo 6.10 Regresyon Modeli Bağımsız Değişken Katsayıları

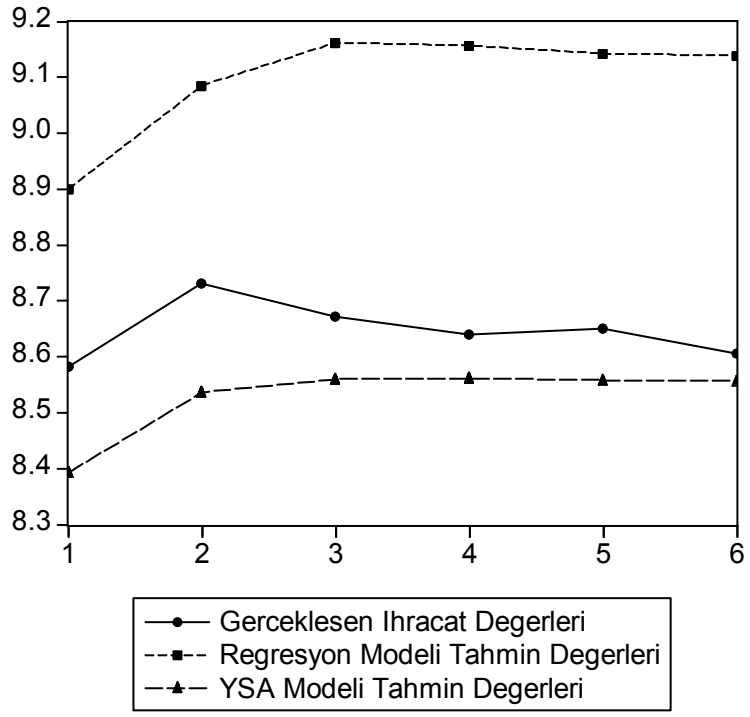
Model	B
(Constant) C_0	-0.4890263106
LOG(IHRAC(-1)) C_1	0. 5799926986
LOG(DOLAR(-1)) C_2	0. 041213104
LOG(SUI(-1)) C_3	0. 8251228541

Tablo 6.10'daki katsayılar kullanılarak 2005:01 ile 2005:06 tarihleri arasında imalat sanayi ihracat değerleri tahmin edilmiştir. Aynı döneme ait yapay sinir ağı tahmin

değerleri ise Alyuda Neurointelligence paket programı kullanılarak elde edilmiş, karşılaştırma yapmak amacı ile gerçekleşen değerler ile her iki modele ait tahmin değerleri Tablo 6.10'da gösterilmiş, Şekil 6.11'de grafik şeklinde gösterilmiştir. Her iki modelde de verilerin doğal logaritması alınarak işlem yapıldığından, tablolar gerçek değerleri değil, logaritmik değerleri göstermektedir.

Tablo 6.11 Tahmin Değerleri ve Gerçekleşen Değerler

Tarih	YSA İhracat Tahmin Değerleri	Regresyon İhracat Tahmin Değerleri	Gerçekleşen İhracat Değerleri
2005:01	8.393213	8.900238	8.581754525
2005:02	8.535269	9.084521	8.73162434
2005:03	8.558553	9.161697	8.671732196
2005:04	8.560449	9.155646	8.640028156
2005:05	8.557618	9.14133	8.649998821
2005:06	8.556058	9.138712	8.606472861



Şekil 6.11 Gerçekleşen Değerler ile Tahmin Edilen Değerlerin Karşılaştırılması

Gerçekleşen değerler ile her iki modele ait tahmin değerleri arasındaki performansların karşılaştırılması Tablo 6.12 ve Tablo 6.13'te gösterilmiştir.

Tablo 6.12 Tahmin Hatasının Doğruluk Ölçütleri

	ME	MSE	RMSE	MAE
Çoklu Regresyon Modeli	-0,45009	0,20943	0,45764	0,45009
Yapay Sinir Ağı Modeli	0,12008	0,01739	0,13186	0,12008

Tablo 6.13 Tahmin Hatasının Yüzde Doğruluk Ölçütleri

	MPE	MSPE	RMSPE	MAPE
Çoklu Regresyon Modeli	-0,05206	0,00280	0,05294	0,05206
Yapay Sinir Ağı Modeli	0,01388	0,00023	0,01523	0,01388

Şekil 6.11, Tablo 6.12 ve Tablo 6.13'te de görüldüğü gibi yapay sinir ağı modeli, çoklu regresyon modeline göre çok daha iyi tahmin değerlerine sahiptir. Bu sonuçlar Bölüm 6.4'teki bulgularımızla örtüşmektedir.

6.6. SONUÇ ve ÖNERİLER

Günümüzde; ekonomilerin dinamik yapıları, gelişen bilgi teknolojileri, karar mekanizmalarında kullanılan verilerin çokluğu, karmaşıklığı ve birbirleri ile olan ilişkileri nedeniyle karar noktasında, geleneksel sistemlerin seri çözüm yöntemlerinden daha fazlasına ihtiyaç duyulmaktadır. Bu nedenle araştırmacılar, insan beynini taklit eden paralel işlem süreçlerine yönlenmektedirler.

Tahmin, gelecekte neyin nasıl olacağının önceden öngörülmesidir ve kesinliği olmayan bir süreçtir. Neredeyse tüm yönetimsel kararlar ileriye yönelik tahminlere bağlıdır, dolayısıyla hem firmalar hem de hükümetler için gelecekteki belirsiz durumların tahmini veya öngörüsü verilecek kararın güvenilirliği açısından önemlidir.

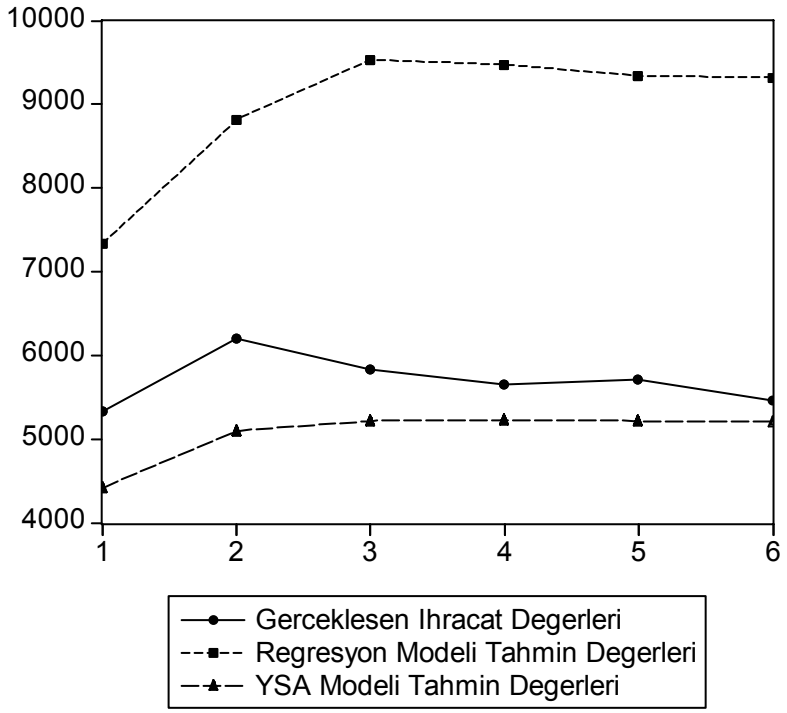
Tahmin modellemesi, karar mekanizmasında önemli rol oynadığından diğer birçok alanda olduğu gibi ekonomi alanında da büyük bir önem taşımakta ve yaygın bir şekilde kullanılmaktadır.

Yapay sinir ağları yöntemi ekonomi alanında ve başka birçok alanda tahmin modellemesi için geliştirilen yöntemlerden biridir. Bu çalışmada tahmin modellemesi için yapay sinir ağları modeli araştırılmıştır. Bu amaç doğrultusunda yapay sinir ağları teorisi detaylı bir şekilde incelenmiş, uygulama alanları ve sık kullanılan bazı YSA mimarileri tanıtılmıştır. Uygulama kısmında ise uygun bulunan bir YSA modeli kullanılarak imalat sanayi ihracat değerleri tahminine yönelik bir tahmin modeli geliştirilmiştir.

Tahmin modellerinde çok sık kullanıldığı, kullanım basitliği, yakınsama hızı ve başarılı sonuçlar vermesi nedeniyle seçilen ileri beslemeli geri yayılım ağı ile kurulan model hem kendi içinde değerlendirilmiş, hem de bir başka tahmin tekniği olan çoklu doğrusal regresyon modeli ile performans karşılaştırılması yapılmıştır.

İlk olarak 1991:01 ile 2004:12 tarihleri arasındaki veriler modellenmiş, ardında da kurulan modellerle 2005 yılının ilk altı ayına ait gerçekleşen değerlerin tahmini yapılmıştır. Sonuçta yapay sinir ağı modelinin gerek modellemede gerekse gerçekleşen değerlerin tahmininde çoklu doğrusal regresyon modeline göre çok

daha iyi sonuçlar vermiştir. Sonuçların kıyaslanması amacıyla elde edilen değerlerin grafik şeklindeki gösterimi Şekil 6.12'de sunulmuştur. Her iki modelden de doğal logaritmik olarak alınan değerler üstel fonksiyon yardımı ile gerçek değerlerine dönüştürülmüştür.



Şekil 6.12 Gerçekleşen Değerler ile Tahmin Edilen Değerlerin Karşılaştırılması

İki model arasında yapılan karşılaştırma sonucunda bir yapay sinir ağı modeli olan ileri beslemeli geri yayılım ağının, tahmin gücü yüksek bir ekonometrik model olan çoklu doğrusal regresyon analizine göre daha esnek ve daha etkili bir tahmin tekniği olduğu görülmüştür.

Ayrıca, yapay sinir ağları yönteminde, çoklu doğrusal regresyon analizindeki gibi verilerin herhangi bir dönüşümüne ihtiyaç duyulmamaktadır. Yapay sinir ağlarının, doğrusal olmayan yapıları nedeniyle doğrusal regresyon analizine göre model kurulumunda avantaj sağladığı ve gerek değişken elemeden gerekse logaritmik dönüşüm yapmadan kurulan modellerde daha başarılı sonuçlar verdiği yapılan uygulamada gösterilmiştir.

Sonuç olarak yapay sinir ağları, istatistiki yöntemlerden farklı olarak, verinin özellikleri ile ilgili sağlanması gereken istatistiki varsayımlarda bulunmadığı gibi algoritma veya matematiksel model geliştirmeye de gerek duymaz. Veri içindeki ilişki kalıplarını ve desenleri tanıyarak öğrenir. Bu nedenle modelin kurulumu ve kullanımı açısından basit, açık ve esnektir. Günümüzde, bu özellikleri ile yapay sinir ağları, tahmin yöntemi olarak pek çok alanda kullanılmakta ve diğer yöntemlere tercih edilmektedir. Problemin yapısına uygun olarak kurulmuş ve eğitilmiş bir yapay sinir ağı, günümüzde kullanılan bir çok tahmin metodundan çok daha iyi sonuçlar verecektir.

Türkiye'deki yapay sinir ağları çalışmaları dünya literatüründeki çalışmalara kıyasla yeterince fazla olmadığı görülmektedir. Araştırmacıların belirli yapay sinir ağları üzerinde yoğunlaşmalarını ve sınıflandırma, tahmin veri kavramlaştırma ve kontrol belirli problemlerin çözümü üzerinde çalışmalarını, ayrıca yeni mimarilerin geliştirilmesi, bulanık mantık ve genetik algoritmalar yardımıyla iyileştirilmesi üzerine teorik çalışmaların yapılmasını önermekteyiz. Amacımız ülkemizin dünya YSA yazınında yeterli çalışma ile temsil edilmesi ve yeni bin yılın teknolojisi olan yapay sinir ağlarının gerek endüstride gerekse savunma sanayi etkin bir şekilde kullanılmasıdır.

KAYNAKÇA

ANDERSON, Dave ve McNeill, George, "**Artıfıcıal Neural Networks Technology**", Rome Laboratory, A DACS State-of-the-Art Report, ELIN: A011, 1992

AKPINAR, Haldun, "**Yapay Sinir Ağları ve Kredi Taleplerinin Değerlendirilmesinde Bir Uygulama**", İU. İşletme Fakültesi Sayısal Yöntemler Anabilim Dalı, 1993

BAŞ, Nuray, "**Yapay Sinir Ağları Yaklaşımı ve bir Uygulama**", Basılmamış Yüksek Lisans Tezi, Mimar Sinan Üniversitesi Fen Edebiyat Fakültesi İstatistik Bölümü, 2006

BELTRATTİ, A., MARGARİTA, S. ve TERNA, P., "**Neural Networks for Economic and Financial Modelling**", International Thomson Computer Press., 1996

CAUDILL, Maureen, "**Neural Network Primar Part 1**", AI Expert, 1987

COZART, Michael Thomas, "**Evaluation of 'The Neural Gas' Network Vector Quantization and Approximation Components**", Master of Science Thesis, The University of Tennessee, Knoxville, 1996

CRONE, Sven F., "**Bussiness Forecasting with Artificial Neural Networks**", presentation of IBF tutorial, Institute of Bussiness Forecasting, 2004

ELMAS, Çetin., 2003, "**Yapay Sinir Ağları: Kuram, Mimari, Eğitim, Uygulama**", Seçkin Yayıncılık, Ankara.

FIESLER, E., "Neural **Network Toplogies, in The Handbook of Neural Computation**" Oxford University Press and IOP Publishing, 1996

FREEMAN James ve SKAPURA, David, **"Neural Networks: Algorithms, Applications and Programming Technicques**", Addison Wesley Publishing Company, 1991

FRÖHLICH, Jochen, "**Neural Net Components in an Object Oriented Class Structure**", (Çevrimiçi) "http://www.nnwj.de/", Fachhochschule Regensburg Department of Computer Science, Germany, 16 Nisan 2006

GURNEY, Kevin, "**An Introduction to Neural Networks**", Department of Human Sciences, Brunnel University, Uxbridge, Middx, 1996

GÜLSEÇEN, Sevinç., "**Yapay Sinir Ağları, İşletme Alanında Uygulanması ve Bir Örnek Çalışma**", Basılmamış Doktora Tezi, İstanbul Üniversitesi Sosyal Bilimler Enstitüsü İşletme Fakültesi Sayısal Yöntemler Anabilim Dalı, 1993

HAND, D.J., "**Discrimination and Classification**", New York: John Wiley & Sons., 1981

HEBB, Donald, "**The Organization of Behavior**", Wile, New York, 1949.

HOSMER, D.W. ve LEMESHOW, S., "**Applied Logistic Regression**", New York: John Wiley & Sons., 1989

JAIN, Anil K., MAO, Jianchang ve MOHIUDDIN, K.M, "**An Artificial Neural Networks: A Tutorial**", IEEE Computer 29(3): 31-44, 1996

KRÖSE, Ben ve SMAGT, Patrick Van Der, "**An Introduction to Neural Networks**", Eight Ed., The University of Amsterdam, 1996

KOHONEN, T.,"**State Of The Art In Neural Computing**", IEEE First International Conference on Neural Networks, 1987

McLACHLAN, G.J., "**Discriminant Analysis and Statistical Pattern Recognition**", New York: John Wiley & Sons., 1992

MEHROTRA, Kishan, MOHAN, Kishan ve RANKA, Sanjay, "**Elements of Artificial Neural Networks**", A Bradford Book, The MIT Pres, Massachusetts Institute of Technology, 1997

MYERS, R.H., "**Classical and Modern Regression with Applications**", Boston: Duxbury Press, 1986

ORHUNBİLGE, Neyran, "**Uygulamalı Regresyon ve Korelasyon Analizi**", İstanbul Üniversitesi İşletme Fakültesi Yayınları, İstanbul, 2002

ÖZTEMEL, Ercan, "**Yapay Sinir Ağları**" Papatya Yayıncılık, 2003

PLUMMER, Eric A., "**Time Series Forecasting With Feed-Forward Neural Networks: Guidelines and Limitations**", Master of Science Thesis in Computer Science, Department of Computer Science, The University of Wyoming, 2000

RAO, Valluru B. "**C++ Neural Networks and Fuzzy Logic**", IDG Boks Worldwide, Inc., 1995

SARAÇ, Tuğba, "**Yapay Sinir Ağları**", Seminer Projesi, Gazi Üniversitesi Endüstri Mühendisliği Anabilim Dalı, 2004

SARLE, Warren S., "**Neural Networks and Statistical Models**", Proceedings of the 19th Annual SAS User Group International Conference, SAS Institute Inc., 1994

SARLE, Warren S., "**Neural Network FAQ**", "ai-faq/neural-nets/part1", (Çevrimiçi) ftp://ftp.sas.com/pub/neural/FAQ.html, Cary, NC, 21 Mart 2006

STADER, Jussi, "**Applying Neural Networks**", Artificial Intelligence Application Institute, University of Edinburg, AIAI-IR-11, 1992

WEISBERG, S., "Applied **Linear Regression**", New York John Wiley & Sons., 1985

WEISS, S.M. ve KULIKOWSKI, C.A., “**Computer Systems That Learn**”, San Mateo, CA: Morgan Kaufmann., 1991

YURTOĞLU, Hasan “**Yapay Sinir Ağlari Metodolojisi Ile Öngörü Modellemesi: Bazi Makroekonomik Değişkenler Için Türkiye Örneği**”, Uzmanlık Tezi, DPT Uzmanlık Tezleri Yayın No: 2683, Ekonomik Modeller ve Stratejik Araştırmalar Genel Müdürlüğü, 2005

EKLER

EK 1: Logaritmik Dönüşüm Uygulanmadan Kurulan Yapay Sinir Ağı Modeli Sonuçları

Tablo EK 1. 1

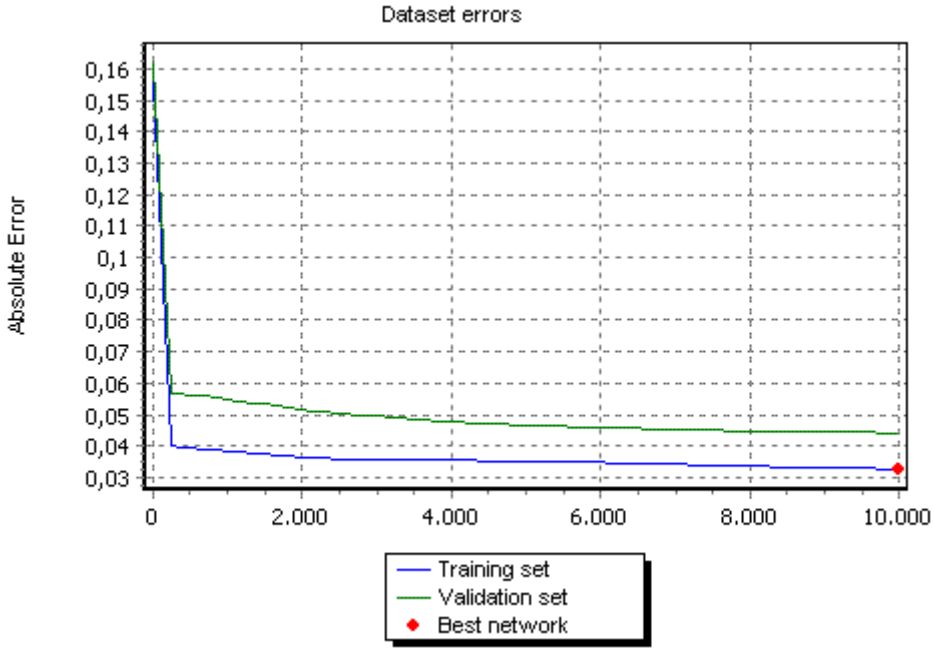
	Target	Output	AE	ARE
Mean:	758,0889	758,0933	0,032595	0,000043
Std Dev:	0,222843	0,216347	0,024118	0,000032
Min:	757,811	757,8687	0,000144	1,90E-07
Max:	758,811	758,7344	0,123847	0,000163
Correlation:	0.983581			
R-Squared:	0.964874			

Tablo EK 1. 2

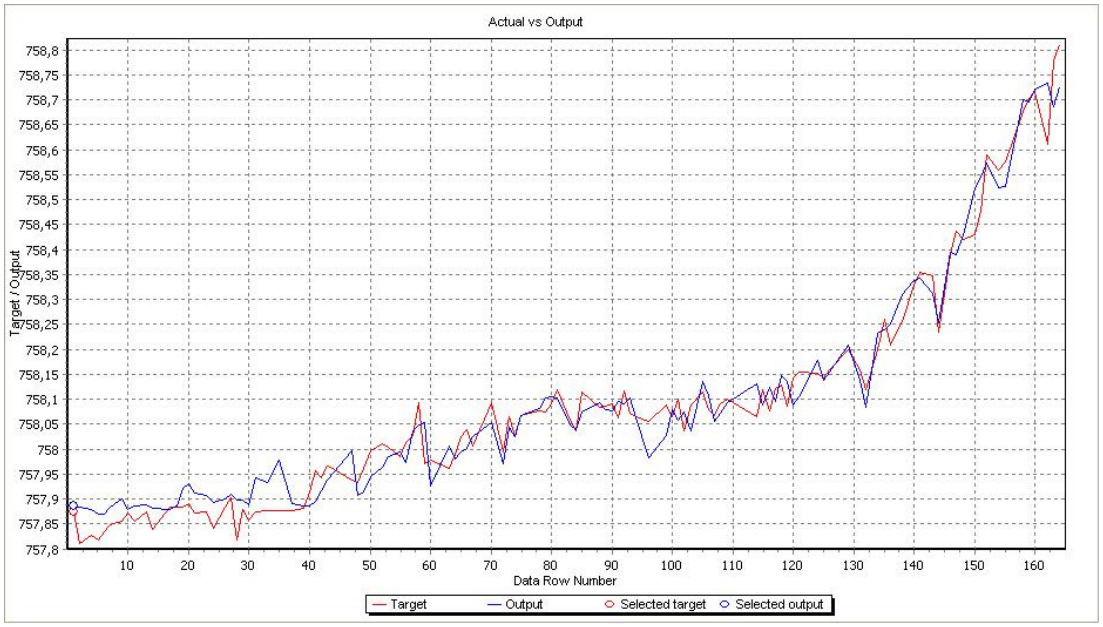
	ME	MSE	RMSE	MAE
Çoklu Regresyon Modeli	0.00000	0.01171	0.10823	0.08404
Yapay Sinir Ağı Modeli	-0.00562	0.00777	0.08814	0.06881
EK 1	-0,00433	0,00164	0,04055	0,03256

Tablo EK 1. 3

	MPE	MSPE	RMSPE	MAPE
Çoklu Regresyon Modeli	-0.00020	0.00021	0.01456	0.01123
Yapay Sinir Ağı Modeli	-0.00076	0.00014	0.01191	0.00921
EK 1	-0,000006	0,000000	0,000053	0,000043



Şekil EK 1. 1



Şekil EK 1. 2

EK 2: Tüm Değişkenler Kullanılarak Kurulan Yapay Sinir Ağı Modeli Sonuçları

Tablo EK 2. 1

	Target	Output	AE	ARE
Mean:	7,530414	7,535011	0,059373	0,007932
Std Dev:	0,463386	0,469402	0,046322	0,006332
Min:	6,629114	6,701179	0,000445	0,000057
Max:	8,603175	8,550879	0,216841	0,028469
Correlation:	0.987096			
R-Squared:	0.974263			

Tablo EK 2. 2

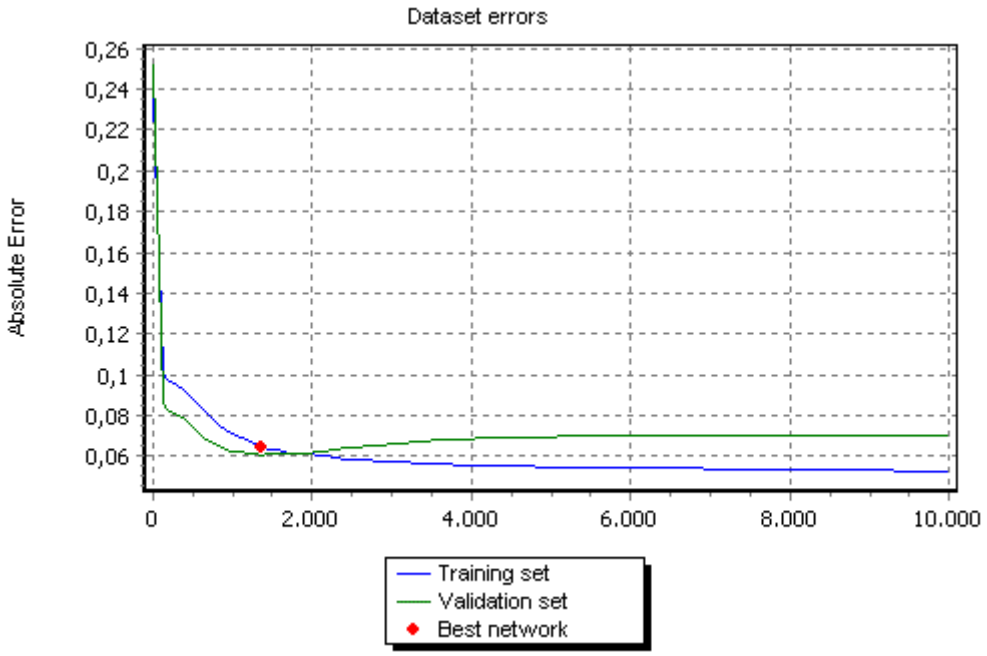
Input column name	Importance, %
İHRACAT	6.74402
DOLAR	58.52289
SUİ	12.02597
USE	19.18515
VERSAAT	3.52196

Tablo EK 2. 3

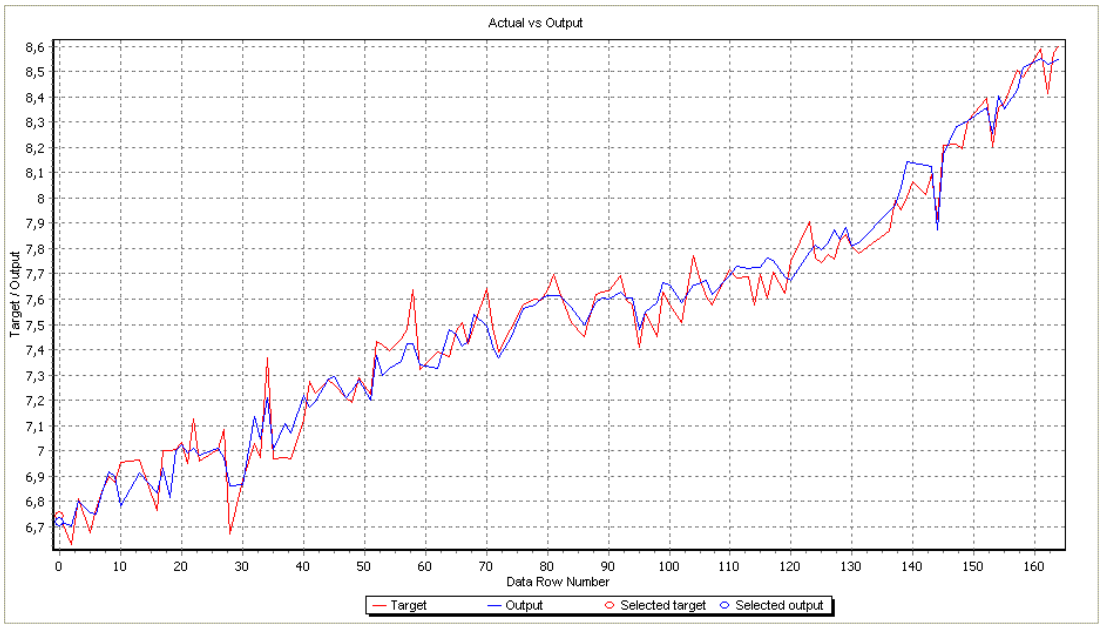
	ME	MSE	RMSE	MAE
Çoklu Regresyon Modeli	0.00000	0.01171	0.10823	0.08404
Yapay Sinir Ağı Modeli	-0.00562	0.00777	0.08814	0.06881
EK 1	-0.00433	0.00164	0.04055	0.03256
EK 2	-0.00460	0.00567	0.07530	0.05937

Tablo EK 2. 4

	MPE	MSPE	RMSPE	MAPE
Çoklu Regresyon Modeli	-0.00020	0.00021	0.01456	0.01123
Yapay Sinir Ağı Modeli	-0.00076	0.00014	0.01191	0.00921
EK 1	-0,000006	0,000000	0,000053	0,000043
EK 2	-0,00062	0,00010	0,01015	0,00793



Şekil EK 2. 1



Şekil EK 2. 2