YAPAY SİNİR AĞLARI İLE BANKA MÜŞTERİSİ BEKLEME SÜRESİ TAHMİNİ

Ümmü Habibe YAZICI

Yüksek Lisans Tezi

Endüstri Mühendisliği Anabilim Dalı
Doç. Dr. Mehmet AKTAN

2010

ATATÜRK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

YÜKSEK LİSANS TEZİ

YAPAY SİNİR AĞLARI İLE BANKA MÜŞTERİSİ BEKLEME SÜRESİ TAHMİNİ

Ümmü Habibe YAZICI

ENDÜSTRİ MÜHENDİSLİĞİ ANABİLİM DALI

ERZURUM
2010

Doç. Dr. Mehmet AKTAN danışmanlığında, Ümmü Habibe YAZICI tarafından hazırlanan bu çalışma 28./.10./2010 tarihinde aşağıdaki jüri tarafından Endüstri Mühendisliği Anabilim Dalı'nda Yüksek Lisans tezi olarak kabul edilmiştir.

Başkan	Doç.Dr.Mehmet AKTAN	İmza
Üye	Y.Doç.Dr. Gökçe AKKAYA	İmza
Üye	Y.Doç.Dr. Bülent ÇAVUŞOĞLU	İmza
Üye	.....	İmza .....
Üye	.....	İmza .....

Yukarıdaki sonucu onaylarım

(İmza)

Prof. Dr. Ömer AKBULUT

Enstitü Müdürü

ÖZET

Y. Lisans Tezi

YAPAY SİNİR AĞLARI İLE BANKA MÜŞTERİSİ BEKLEME SÜRESİ TAHMİNİ

Ümmü Habibe YAZICI

Atatürk Üniversitesi
Fen Bilimleri Enstitüsü
Endüstri Mühendisliği Anabilim Dalı

Danışman: Doç. Dr. Mehmet AKTAN

Bu çalışmada amaç, banka müşterilerinin bekleme sürelerinin tahmini için yapay sinir ağları yönteminin kullanımının araştırılmasıdır. Bu amaç doğrultusunda yapay sinir ağları genel hatlarıyla tanıtılmış olup, yapıları incelenmiştir. Yapay sinir ağlarının temel öğrenme kuralları açıklanmış, bir banka şubesinde toplanan veriler kullanılarak şubeye gelen müşterilerin tahmini bekleme sürelerinin elde edilmesinde yapay sinir ağlarının uygulaması gerçekleştirilmiştir.

2010, 64 sayfa

Anahtar Kelimeler: Yapay sinir ağları, banka, bekleme süresi

ABSTRACT

MS Thesis

BANK CUSTOMER WAITING TIME ESTIMATION with
ARTIFICIAL NEURAL NETWORKS

Ümmü Habibe YAZICI

Atatürk University
Graduate School of Natural and Applied Sciences
Department of Industrial Engineering

Supervisor: Assoc. Prof. Mehmet AKTAN

The aim in this study is to explore the use of artificial neural networks for the estimation of bank customer waiting time. For this purpose, artificial neural networks were introduced in general terms, and their structures were investigated. Basic learning rules of artificial neural networks were explained and they were applied for the estimation of a bank's customers' waiting times by using the data obtained from the bank.

2010, 64 pages

Keywords: Artificial neural networks, bank, waiting time

TEŞEKKÜR

Bu tezin hazırlanması sırasında öncelikle danışmanlık yaparak değerli bilgi ve desteğini esirgemeyen sayın Doç. Dr. Mehmet AKTAN'a ve desteği için eşim Kürşat YAZICI'ya çok teşekkür ederim.

Ümmü Habibe YAZICI
Eylül 2010

İÇİNDEKİLER

SİMGELER ve KISALTMALAR DİZİNİ

d	Geriye Yayılacak Hata Terimi
H	Öğrenme Katsayısı
K	Katman Sayısı
μ	Momentum
N	Devir Sayacı
p, q	Hücre numarası
w	Ağırlık Değeri
Φ(I)	Ağın Bulduğu Sonuç
X	Girdi

Kısaltmalar

ÇKA	Çok Katmanlı Algılayıcı
GYA	Geri Yayılım Algoritması
İE	İşlemci Elemanlar
MLP	Multi Layer Perceptron
YSA	Yapay Sinir Ağı

ŞEKİLLER DİZİNİ

ÇİZELGELER DİZİNİ

1. GİRİŞ

1.1. Yapay Sinir Ağları

Yapay Sinir Ağları (YSA), beynin fizyolojisinden yararlanılarak oluşturulan bilgi işleme modelleridir. Yapay sinir ağları, basit biyolojik sinir sisteminin çalışma şeklini simüle etmek için tasarlanan programlardır (Yurtoğlu 2005). Ayrıca, yapay sinir ağları için genel bir tanım vermek gerekirse; "Yapay sinir ağları, en kısa ve basit bir şekilde, bir örnek kümesi yardımıyla parametrelerin uyarlanabilmesini sağlayacak bir matematiksel formül için yazılan bilgisayar programı" olarak tanımlanabilir (Ersöz vd. 2008).

Literatürde yüzden fazla yapay sinir ağı modeli vardır. Bazı bilim adamları, beynin güçlü düşünme, hatırlama ve problem çözme yeteneklerini bilgisayara aktarmaya çalışmışlardır. Bazı araştırmacılar ise, beynin fonksiyonlarını kısmen yerine getiren model oluşturma konusunda çalışmalarda bulunmuşlardır.

Yapay sinir ağlarının öğrenme özelliği, araştırmacıların dikkatini çeken en önemli özelliklerden birisidir. Çünkü herhangi bir olay hakkında girdi ve çıktılar arasındaki ilişkiyi, doğrusal olsun veya olmasın, elde bulunan mevcut örneklerden öğrenerek daha önce hiç görülmemiş olayları, önceki örneklerden çağrışım yaparak ilgili olaya çözümler üretebilme özelliği yapay sinir ağlarındaki zeki davranışın da temelini teşkil eder.

1943 yılında bir nörobiyolojist olan Warren McCulloch ve bir istatistikçi olan Walter Pitts, "Sinir Aktivitesindeki Düşüncelere Ait Bir Mantıksal Hesap" başlıklı bir makale ile ilk dijital bilgisayarlara ışık tutmuştur. John Von Neumann bu makaleyi, "elektronik beyinler" için bir kopya olarak görmüştür. Yapay zeka alanındaki araştırmacılar içerisinde istisnai bir yeri olan Marvin Minsky, bu makaleden aldığı ilhamla makroskobik zeka fikrini ortaya atmış ve uzman sistemlerin doğmasına neden olmuştur.

Bronx Yüksek Bilim Okulu'ndan Frank Rosenblatt, gözün hesaplamaları ile ilgilenmiştir. Bu bilim adamları, öğrenmenin ve zekanın herhangi bir özelliğinin simülasyonunda bilgisayarların aktif olarak nasıl kullanılabileceğini, 1956 yılında düzenlemiş oldukları ilk yapay zeka konferansında tartışmışlardır.

1959'da, Stanford üniversitesinden Bernard Widrow, basit nöron benzeri elemanlara dayanan ve "ADALINE" (Adaptive Linear Neuron) olarak adlandırılan bir adaptif lineer eleman geliştirmiştir. ADALINE ve iki tabakalı biçimi olan "MADALINE" (Multiple Adaline); ses tanıma, karakter tanıma, hava tahmini ve adaptif kontrol gibi çok çeşitli uygulamalar için kullanılmıştır. Daha sonraları ADALINE, ayrık bir çıkış yerine sürekli bir çıkış üretmek için geliştirilmiştir. Widrow, telefon hatları üzerindeki ekoları elimine etmeye yarayan adaptif filtreleri geliştirmede, adaptif lineer eleman algoritmasını kullanmıştır. Bununla ilk defa yapay sinir ağları, gerçek bir probleme uygulanmıştır.

Helsinki Teknik Üniversitesi'nden Teuvo Kohonen, 1970'lerin ilk yıllarında adaptif öğrenme ve birleşik hafızalar üzerine temel çalışmalar yapmış ve bu çalışmalar ile danışmansız öğrenme metotlarının gelişmesine ışık tutmuştur.

Minsky ve Papert'in Perseptron isimli kitaplarında, yapay sinir ağlarının temel olarak ilgi çekici konular olmadığını belirtmeleri, birçok araştırmacının bu alanda çalışmaktan vazgeçmelerine sebebiyet vermiştir. YSA konusunda çalışmaya devam eden Grossberg, yapay sinir ağları modellerini yapılandırmak için nörolojik verinin kullanılması, algı ve hafıza için yapay sinir ağları tabanlı mekanizmaların önerilmesi, belirgin eşitliklerle bütünleşen bir sinaptik model için, ilişkilendirici bir kural üzerinde çalışmıştır.

1982 yılında ilgi çeken bir başka gelişme, moleküler biyolojiden beyin kuramcılığına geçiş yapan bir model, Caltech fizikçisi Hopfield tarafından sunulmuştur. Kendi adıyla anılan bir ağ yapısı mevcuttur ve birçok alana uygulanmıştır. 1987 yılında yapılan ilk yapay sinir ağları sempozyumundan sonra, yapay sinir ağları uygulamaları

yaygınlaşmıştır. Günümüzde, yapay sinir ağları ile ilgili araştırmalar yapan çok sayıda bilim adamı ve araştırma grupları vardır (Aksungur 2009).

Bu çalışmada yapay sinir ağlarının tahmin özelliği, bir banka müşterisinin gün, bekleyen kişi sayısı, kartlı veya kartsız olma, bireysel ya da şirket müşterisi olma durumlarına göre ne kadar süre bekleyeceğinin öngörülmesinde kullanılmıştır.

2. KURAMSAL TEMELLER

2.1. Yapay Sinir Ağlarının Genel Özellikleri

Çok değişik amaçlar için çok sayıda sinir ağı geliştirilmiştir. Yapısı, çalışması ve işlem prensibi bakımından farklılık göstermekle birlikte bazı özellikleri ortaktır. Genel anlamda YSA tümüyle birbirine bağlantılı pek çok sayıda sinyal ya da bilgi işleme birimlerinden oluşmuş bir hesaplama sistemidir ve aşağıdaki özelliklere sahiptir.

a. Paralel çalışma: Yapay sinir ağlarında tüm işlem elemanları eş zamanlı çalıştıkları için çok hızlı çıktı üretirler.

b. Doğrusal olmama: YSA'nın temel işlem elemanı olan hücre doğrusal değildir. Dolayısıyla hücrelerin birleşmesinden meydana gelen YSA da doğrusal değildir ve bu özellik tüm ağa yayılmış durumdadır. Bu özelliği ile YSA, doğrusal olmayan karmaşık problemlere çözüm getirmektedir.

c. Genelleme: YSA, ilgilendiği problemi öğrendikten sonra eğitim sırasında karşılaşmadığı test örnekleri için de belirtilen tepkiyi üretme kabiliyetine sahiptir. Örneğin, karakter tanıma amacıyla eğitilmiş bir YSA, bozuk karakter girişlerinde de doğru karakteri verir. Eğitilmiş bir ağa girişin sadece bir kısmı verilse bile, ağ hafızadan bu girişe en yakınını seçerek tam bir giriş verisi alıyormuş gibi kabul eder ve buna uygun bir çıkış değeri üretir. Veri YSA'ya, eksik, bozuk veya daha önce hiç karşılaşmadığı şekilde verilse bile, ağ kabul edilebilir en uygun çıkışı üretecektir. Bu özellik ağın genelleştirme özelliğidir (Yıldız 2006).

Yapay sinir ağlarının hesaplama ve bilgi işleme gücünü, paralel dağılmış yapısından, öğrenebilme ve genelleme yeteneğinden aldığı söylenebilir. Genelleme, eğitim ya da öğrenme sürecinde karşılaşılmayan girişler için de yapay sinir ağının uygun tepkileri

üretmesi olarak tanımlanır. Bu üstün özellikleri, yapay sinir ağının karmaşık problemleri çözebilme yeteneğini gösterir (Ergezer 2003).

d. Öğrenme: Yapay sinir ağları insan zekası gibi örneklerle eğitilirler. Eğitme algoritmaları YSA'nın ayrılmaz bir parçasıdır. Eğitme algoritması eldeki problemin özelliğine göre öğrenme kuralını YSA'na nasıl adapte edeceğimizi belirtir. Ağlar ne kadar çok örnekle eğitilirse problem üzerindeki teşhisi o kadar doğru olur (Kaçar Durmuş ve Meriç 2005). Olayları öğrenerek benzer olaylar karşısında benzer kararlar vermeye çalışırlar. Böylelikle kendisine gösterilen örneklerden genellemeler yaparak daha önce görmediği örnekler hakkında bilgiler üretebilirler.

e. Bilginin saklanması: YSA'larda bilgi ağın bağlantılarında saklanmaktadır. Diğer programlarda ise bir veri tabanında ya da programın içerisinde gömülü olarak bulunur.

f. Hata toleransı: YSA'lar, çok sayıda işlemci elemanların bağlantısının paralel dağılmış olduğu bir yapıya sahiptir ve ağın sahip olduğu bilgi, ağdaki tüm bağlantılara dağılmıştır. Giriş verisinde bulunabilecek herhangi bir gürültü, bütün ağırlıklar üzerine dağıtıldığından dolayı, gürültü etkisi tolere edilebilir. Girişlerde eksik bir bilgi sistemin tamamının çalışmasını engellemez. Geleneksel yöntemlere göre hatayı tolere etme yetenekleri daha fazladır.

g. Uyarlanabilirlik: YSA ağırlıkları, uygulanan probleme göre değiştirilir. Yani, belirli bir problemi çözmek amacıyla eğitilen YSA, problemdeki değişimlere göre tekrar eğitilebilir. Değişimler devamlı ise gerçek zamanda da eğitime devam edilebilir. Bu özelliği ile YSA, uyarlamalı örnek tanıma, işaret işleme, sistem tanımlama ve denetim gibi alanlarda etkin olarak kullanılır.

h. Kendi ilişkisini oluşturma: Yapay sinir ağları verilere göre kendi ilişkilerini kendisi oluşturabilir. Bünyesinde sabit bir denklem içermez.

i. Sınırsız sayıda değişken ve parametre kullanımı: Yapay sinir ağları sınırsız sayıda değişken ve parametre ile çalışabilir. Bu sayede çok başarılı tahmin ve genel çözümler üretebilmektedir.

j. Algılamaya yönelik olaylarda kullanılabilirlik: Yapay sinir ağları daha çok algılamaya dönük bilgileri işlemede kullanılırlar. Bilgiye dayalı işlemlerde genellikle uzman sistemler kullanılır. Bazı durumlarda bu iki sistem birleştirilerek daha başarılı sonuçlar üreten bir sistem elde edilebilir. Şekil 2.1'de uzman sistemler ile yapay sinir ağlarının karşılaştırılması yapılmıştır.

Karakteristik	Uzman Sistemler	Yapay Sinir Ağları
Yaklaşım	Sembolik	Sayısal
Nedensellik	Mantıksal	İlişkisel
Operasyon	Mekanik	Biyolojik, doğaya uygun
Açıklama	Mümkün	Mümkün değil
İşlem	Sıralı	Paralel
Sistem	Kapalı	Kendi kendine ayarlanabilir
Doğrulama ve onaylama	Yavaş	Hızlı
Neyle sürülür?	Bilgi / Tecrübe ile	Veriler ile
Tamiri	Zor	Kolay

Şekil 2.1. Uzman Sistemler ile YSA'ların Karşılaştırılması

k. Dereceli bozulma: Hatalara karşı toleranslı oldukları için sistemin bozulması da dereceli olur. Yani klasik programlarda sistemde bir hata var ise sistem tamamen çalışamaz duruma geçer, yorum yapamayacağı için kısmi de olsa bilgi üretemez. Fakat YSA'lar eldeki verilerle, sağlam olan hücrelerle bilgi üretmeye çalışırlar (Yıldız 2006).

l. Nümerik bilgi ile çalışma: Yapay sinir ağları sadece nümerik bilgi ile çalışırlar. Sembolik ifadeler ile gösterilen bilgilerin nümerik gösterime çevrilmeleri gerekmektedir (Öztemel 2003).

Biyolojik sinir ağlarının sinir hücrelerinden oluşması gibi, yapay sinir ağları da yapay sinir hücrelerinden meydana gelmektedir. Yapay sinir hücreleri ayrıca düğüm (node), birim (unit) veya işlemci eleman (processing unit) olarak da adlandırılmaktadır. Bir yapay sinir ağı, birbiriyle bağlantılı çok sayıda yapay sinir hücresinden meydana gelmektedir. Şekil 2.2'de biyolojik sinir ağı ve yapay sinir ağının karşılaştırması yapılmıştır. Yapay sinir hücreleri biyolojik sinir hücrelerinin basit bir modelidir.

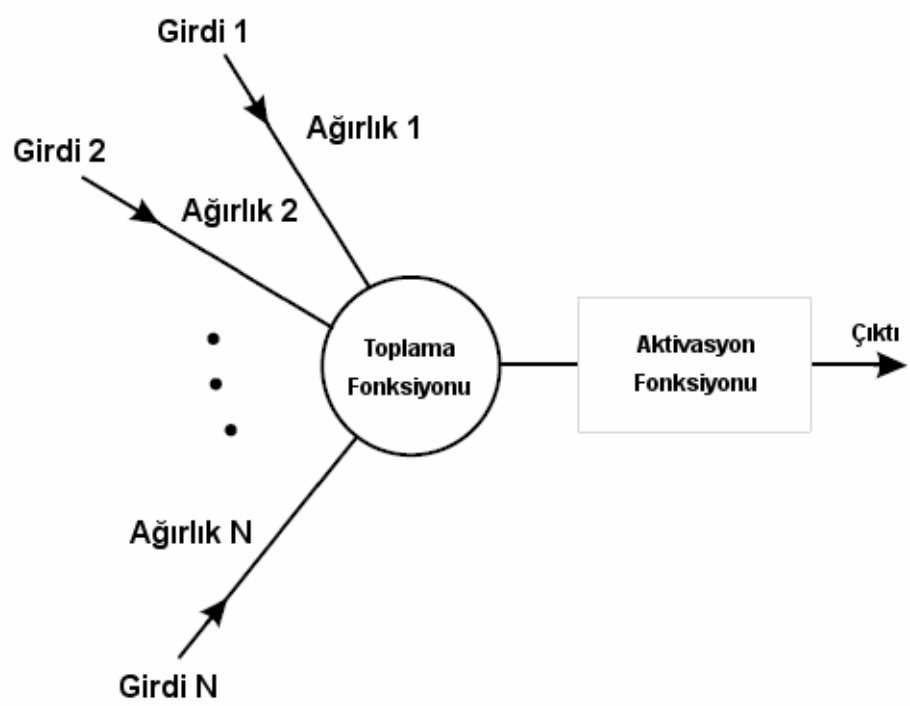
Biyolojik Sinir Ağı	Yapay Sinir Ağı
Sinir Sistemi	Sinirsel Hesaplama Sistemi
Sinir Hücresi (Nöron)	İşlemci Eleman (Yapay Sinir Hücresi, Düğüm)
Sinaps	İşlemci elemanlar arasındaki bağlantı ağırlıkları
Dendrit	Toplama Fonksiyonu
Hücre Gövdesi	Aktivasyon Fonksiyonu
Akson	İşlemci Eleman çıktısı

Şekil 2.2. Biyolojik Sinir Ağı ve Yapay Sinir Ağının Karşılaştırılması

Yapay sinir ağlarının içinde bulunan tüm sinir hücreleri bir veya birden fazla girdi alırlar ve tek bir çıktı verirler. Bu çıktı yapay sinir ağının dışına verilen bir çıktı olabileceği gibi başka bir yapay sinir hücresine girdi olarak da verilebilir. Yapay sinir hücresinin yapısı Şekil 2.3'de gösterilmektedir.

Bir yapay sinir hücresi genel olarak beş temel bileşenden oluşmaktadır.

- Girdiler
- Ağırlıklar
- Toplama fonksiyonu
- Aktivasyon fonksiyonu
- Çıktı



Şekil 2.3. Yapay Sinir Hücresinin Yapısı

2.2. Yapay Sinir Ağlarının Yapısı

2.2.1. Girdiler

Girdiler, bir yapay sinir hücresine gelen bilgilerdir. Bu bilgiler dış ortamlardan ya da diğer sinir hücrelerinden gelebilir. Dış ortamlardan gelen bilgiler, ağın öğrenmesi istenen örnekler tarafından belirlenmektedir.

Basit bir yapay nöron yapısında ilk adım girdi değerlerinin ağırlıklandırılmasıdır. Bir nöron genellikle eş anlı olarak birçok sayıda girdi alır. Her girdinin kendi nispi ağırlığı vardır. Bazı girdiler diğerlerine göre daha önemli hale gelebilirler ve nispi ağırlık değerleri daha yüksek olur. Böylelikle işlem elemanlarının bir sinirsel tepki üretmesi işleminde daha fazla etkili olurlar, yani ağırlıklar girdinin bağlantı gücünün bir ölçüsüdür (Ekinci vd. 2008).

2.2.2. Ağırlıklar

Ağırlıklar, gelen bilgilerin hücre üzerindeki etkisini belirleyen değerlerdir. Bilgiler, bağlantılar üzerindeki ağırlıklar üzerinden hücreye girmekte ve ağırlıklar yapay sinirde girdi olarak kullanılacak değerlerin göreceli kuvvetini (matematiksel katsayısını) göstermektedirler. Yapay sinir ağı içinde girdilerin hücreler arasında iletimini sağlayan tüm bağlantıların farklı ağırlık değerleri bulunur. Böylelikle ağırlıklar her işlemci elemanın her girdisi üzerinde etki yapmış olur. Ağırlıklar değişken veya sabit değerler olabilirler.

2.2.3. Toplama fonksiyonu

Toplama fonksiyonu, hücreye gelen net girdiyi hesaplayan fonksiyondur ve genellikle girişlerin kendi ağırlıklarıyla çarpımının toplamı;

$$s = \sum x_i w_i$$

Biçiminde ifade edilir.

Toplama fonksiyonunda genelde ağırlıklı toplam kullanılır. Ağırlıklı toplam, nöron girdilerinin sinaptik bağlantılar üzerindeki ağırlıkları ile çarpılarak bulunur (Burmaoğlu 2009). Yapay sinir ağının yapısına göre toplama fonksiyonu, maksimum, minimum, çarpım veya çeşitli normalizasyon işlemlerinden birisi olarak da ifade edilebilir. Bir

problem için en uygun toplama fonksiyonu çeşidini bulmak için herhangi bir formül yoktur. Toplama fonksiyonu genellikle deneme yanılma yoluyla bulunmaktadır. Ayrıca bir yapay sinir ağındaki bütün işlemci elemanların aynı toplama fonksiyonuna sahip olması gibi bir zorunluluk da yoktur. Bazen aynı yapay sinir ağı içindeki işlemci elemanların bazıları aynı toplama fonksiyonunu, diğerleri ise başka fonksiyonları kullanabilirler. Bu tamamen tasarımcının kendi kararına bağlıdır.

2.2.4. Aktivasyon fonksiyonu

Aktivasyon fonksiyonu, toplama fonksiyonundan gelen girdiyi işleyerek yapay sinir hücresinin çıkışını belirler. Transfer fonksiyonu olarak da adlandırılan aktivasyon fonksiyonu çeşitli tiplerde ve genellikle doğrusal olmayan bir fonksiyondur. Doğrusal fonksiyonların tercih edilmemesinin nedeni, doğrusal fonksiyonlarda girdi ile çıktının doğru orantılı olmasıdır. Bu durum ilk yapay sinir ağları denemelerinin başarısızlıkla sonuçlanmasının temel nedenidir.

Uygun aktivasyon fonksiyonunun seçimi tasarımcının farklı fonksiyonları denemeleri sonucunda belirlenmektedir. Ancak çok katmanlı perceptron gibi bazı modeller aktivasyon fonksiyonunun, türevi alınabilir bir fonksiyon olmasını şart koşmaktadır. Ayrıca fonksiyonun seçimi, yapay sinir ağının verilerine ve neyi öğrenmesinin istendiğine de bağlıdır. Aktivasyon fonksiyonu olarak en çok kullanılanlar sigmoid fonksiyon ve hiperbolik tanjant fonksiyonlarıdır.

Aktivasyon fonksiyonu, toplama fonksiyonundan gelen girdiyi dönüştürerek istenilen değerler arasında sınırlandırmaktadır. Bu değerler kullanılan aktivasyon fonksiyonun tipine göre genellikle [0,1] veya [-1,1] arasındadır. Bu değer aktivasyonun fonksiyonunun, dolayısıyla yapay sinir hücresinin çıktı değeri olarak ya dış ortama ya da girdi olarak başka bir yapay sinir hücresine iletilmektedir.

Aktivasyon fonksiyonu işlemi öncesinde, sisteme tekdüze (uniform) dağılmış bir rassal hata eklenebilmektedir. Bu rastsal hatanın kaynağı ve büyüklüğü sistemin öğrenme

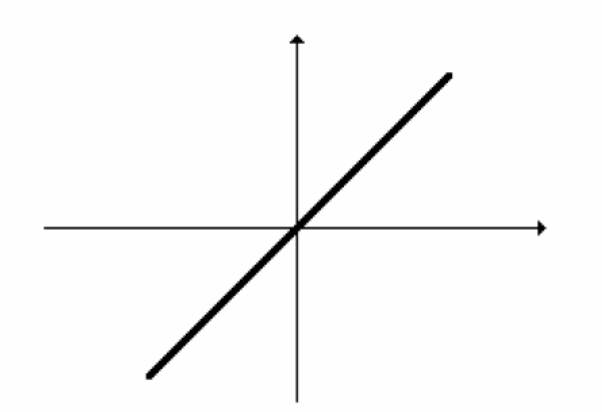
sürecinde belirlenir ve sebebi, insan beyninin işlevinin, içinde bulunduğu ortamın koşullarından etkilenmesidir. Örneğin ortamın soğuk/sıcak olmasından insan beyni etkilenmektedir. Bu nedenle yapay sinir ağları literatüründe rastsal hata ekleme işlemi "sıcaklık (temperature)" olarak da adlandırılmaktadır. Ancak günümüzde rastsal hata işlevi tam olarak kullanılmamakta ve hala bir araştırma süreci içinde bulunmaktadır. Ayrıca bazı yapay sinir ağlarında, aktivasyon fonksiyonunun çıktısı üzerinde başka işlemler, ölçeklendirme ve sınırlandırma yapılabilmektedir. Yapay sinir hücrelerinde yaygın olarak kullanılan çeşitli aktivasyon fonksiyonları aşağıda verilmiştir.

2.2.4.a. Doğrusal aktivasyon fonksiyonu

Doğrusal problemlerin çözümünde kullanılan bu fonksiyon, gelen net girdileri doğrudan hücre çıkışı olarak vermektedir. Matematiksel olarak

$$F(s) = s$$

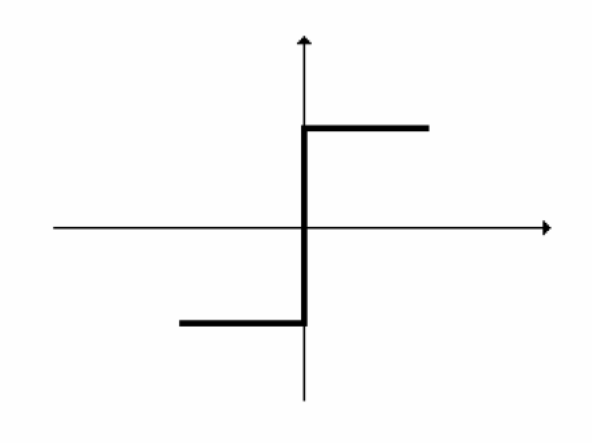
Şeklinde tanımlanmaktadır. Şekilsel gösterimi ise; Şekil 2.4'de görüldüğü gibidir.



Şekil 2.4. Doğrusal Aktivasyon Fonksiyonu'nun Şekilsel Gösterimi

2.2.4.b. Adım fonksiyonu

Gelen net girdi değerinin belirlenen bir eşik değerinin altında ya da üstünde olmasına göre hücrenin çıktısı 1 veya 0 değerlerini almaktadır. Adım fonksiyonunun şekilsel gösterimi Şekil 2.5'de görüldüğü gibidir.



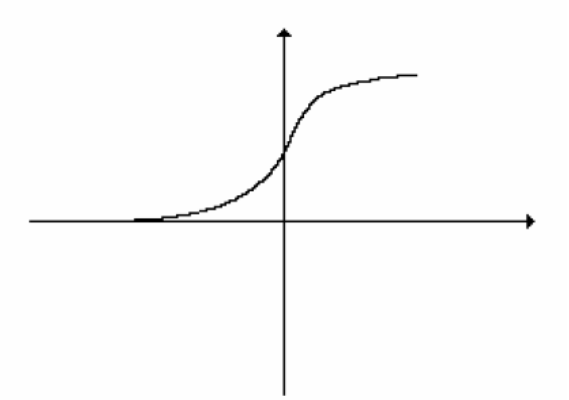
Şekil 2.5. Adım Fonksiyonu'nun Şekilsel Gösterimi

2.2.4.c. Sigmoid aktivasyon fonksiyonu

Sigmoid aktivasyon fonksiyonu, türevi alınabilir, sürekli ve doğrusal olmayan bir fonksiyon olması nedeniyle uygulamada en çok kullanılan aktivasyon fonksiyonudur. Bu fonksiyon, net girdinin her değeri için 0 ile 1 arasında bir değer üretmektedir ve formülü şu şekildedir.

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

Sigmoid fonksiyonun şekilsel gösterimi Şekil 2.6'da görüldüğü gibidir.



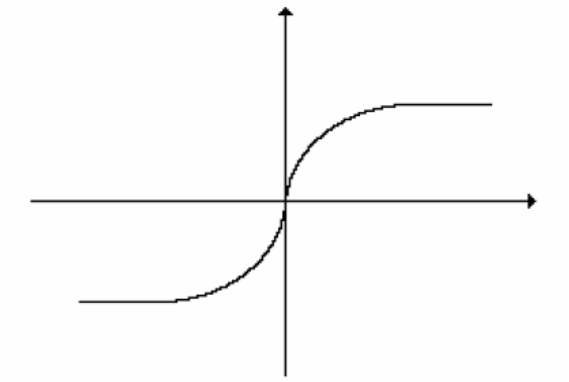
Şekil 2.6. Sigmoid Aktivasyon Fonksiyonu'nun Şekilsel Gösterimi

2.2.4.d. Hiperbolik tanjant fonksiyonu

Hiperbolik tanjant fonksiyonu, gelen net girdinin tanjant fonksiyonundan geçirilmesi ile hesaplanmaktadır ve sigmoid aktivasyon fonksiyonunun farklı bir çeşididir. Sigmoid aktivasyon fonksiyonunda çıktı 0 ile 1 arasında bir değer alırken, hiperbolik tanjant fonksiyonunda çıktı -1 ile 1 arasındadır ve şu şekilde hesaplanır.

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

Bu fonksiyonun şekilsel gösterimi Şekil 2.7'de görüldüğü gibidir.

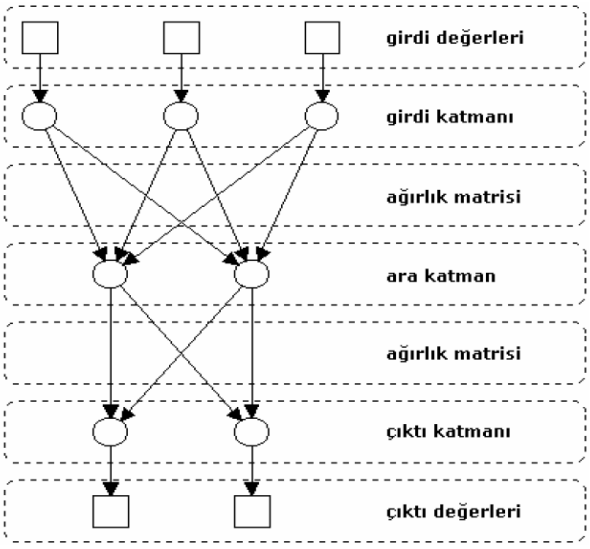


Şekil 2.7. Hiperbolik Tanjant Fonksiyonu'nun Şekilsel Gösterimi

2.2.5. Hücrenin çıktısı

Aktivasyon fonksiyonu tarafından belirlenen çıktı değeridir. Bu değer ya başka bir yapay sinir hücresine girdi olarak ya da dış ortama gönderilmektedir. Bir işlemci elemanın birden fazla girdisi olmasına rağmen tek bir çıktısı olmaktadır.

Yapay sinir ağlarını bir kez daha kısaca anlatacak olursak; Yapay sinir hücreleri bir araya gelerek yapay sinir ağını meydana getirmektedir. Yapay sinir hücrelerinin bir araya gelmesi rastgele değildir. Yapay sinir hücrelerinin genellikle birbiriyle bağlantılı 3 katman halinde ve her katman içinde birbirine paralel olacak şekilde bir araya gelerek ağı oluşturdukları görülür. Bu 3 katman, girdi katmanı (input layer), ara katman (hidden layer) ve çıktı katmanı (output layer) olarak adlandırılır. Girdi katmanı, dış ortamdan bilgileri alarak ara katmanlara iletmekle görevlidir. Bazı yapay sinir ağlarında bu katmanda herhangi bir bilgi işleme olmamaktadır. Ara katman, girdi katmanından gelen bilgileri işleyerek çıktı katmanına iletmekle görevlidir. Gizli katman olarak da adlandırılan ara katman birden fazla olabilmektedir. Ara katmanlar çok sayıda yapay sinir hücresi içermektedirler ve bu hücreler yapay sinir ağı içindeki diğer hücrelerle bağlantılıdırlar. Çıktı katmanı, ara katmandan gelen işlenmiş bilgileri dış ortama aktarmakla görevlidir. Yapay sinir ağı mimarisi Şekil 2.8'de gösterildiği gibidir.



Şekil 2.8. Tipik 3 Katmanlı İleri Beslemeli Yapay Sinir Ağı Mimarisi

2.3. Yapay Sinir Ağlarının Eğitimi ve Testi

Yapay sinir ağlarında işlemci elemanlar arasındaki bağlantıların ağırlık değerlerinin değiştirilmesi işlemine "ağın eğitilmesi" denilmektedir.

Başlangıçta rastgele atanan bu ağırlık değerleri, ağa gösterilen örneklerle değiştirilmektedir. Amaç, ağa gösterilen örnekler için doğru çıktıları üretecek ağırlık değerlerinin belirlenmesidir. Yapay sinir ağının eğitilmesinde kullanılan girdi ve çıktı çiftlerinden oluşan verilerin tümüne "eğitim seti" adı verilmektedir.

Yapay sinir ağlarının eğitim süreci, belli kurallar çerçevesinde olmaktadır. Bu kurallara öğrenme kuralları adı verilmektedir. Ağırlıkların değiştirilmesi öğrenme kurallarına göre yapılır. 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 evrenden gelen girdi-çıktı örneklerini doğru sınıflandırabilme yeteneği olarak tanımlanır. Ağın bu genelleştirme özelliğine kavuşması işlemine "ağın öğrenmesi" denilir.

Yapay sinir ağlarında öğrenme iki aşamada gerçekleşir. Birinci aşamada ağa gösterilen örnek için ağın üreteceği çıktı belirlenir. Bu çıktı değerinin doğruluk derecesine göre, ikinci aşamada ağın bağlantılarının sahip olduğu ağırlıklar değiştirilmektedir. Ağın çıktısının belirlenmesi ve ağırlıkların değiştirilmesi öğrenme kuralına bağlı olarak farklı şekillerde olmaktadır.

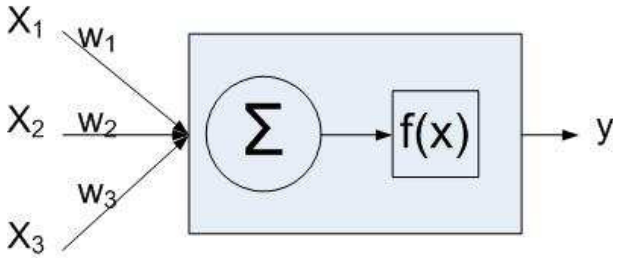
Bir yapay sinir ağının eğitiminin tamamlanmasının ardından, ağın öğrenip öğrenmediğini (performansını) ölçmek için denemeler yapılarak ağın test edilmesi gerekmektedir. Bir ağı test etmek için ağın eğitimi sırasında görmediği, yani veri setinden test amaçlı olarak ayrılan örnekler kullanılır ve bu örnekler "test seti" adını alır. Test işleminde ağın ağırlık değerleri değiştirilmemektedir. Örnekler ağa gösterilmekte ve ağ eğitimi sırasında 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.

Eğitim ve test setleriyle ilgili temel sorun, yeterli eğitim ve test verisi miktarının ne olması gerektiğidir. Sınırsız sayıda veri bulunabilmesi durumunda, yapay sinir ağı mümkün olduğunca çok veriyle eğitilmelidir.

Eğitim verisinin yeterli olup olmadığı konusunda emin olmanın yolu, eğitim verisinin miktarının arttırılarak, bunun ağın performansında bir değişiklik yaratıp yaratmadığına bakmaktır. Ancak bunun mümkün olmadığı durumlarda yapay sinir ağının eğitim ve test verileri üzerindeki performansının yakın olması da verilerin yeterli olduğuna ilişkin bir gösterge olarak kabul edilebilir. Bununla birlikte eğitim setinin içermesi gereken veri miktarı değişik yapay sinir modellerine ve özellikle problemin gösterdiği karmaşıklığa göre farklılık göstermektedir (Baş 2006).

2.3.1. Yapay sinir hücresinin çalışmasına bir örnek



Şekil 2.9. Yapar Sinir Hücresi

Örneğimizde;

$$X_1 = 0.9, \quad X_2 = 1.4, \quad X_3 = 2.6 \qquad \text{ve} \qquad w_1 = 0.2, \quad w_2 = 0.4, \quad w_3 = 0.6$$

olsun.

Bu durumda sigmoid olarak belirlemiş olduğumuz aktivasyon fonksiyonuna girecek değer:

$$\sum = 0.9x0.2 + 1.4x0.4 + 2.6x0.6 = 2.3$$

olacaktır. Hücremizin ürettiği sonuç olan y ise:

$$y = f(2.3) = \frac{1}{1+e^{-2.3}} \cong 0.90$$

olur (Yıldız 2006)**.**

2.4. Yapay Sinir Ağlarının Sınıflandırılması

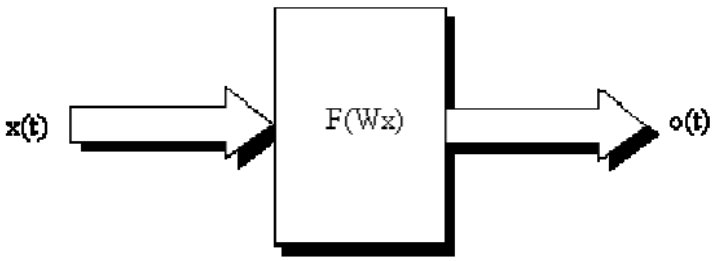
Her bir sinir hücresi arasındaki bağlantıların yapısı ağın yapısını belirler. İstenilen hedefe ulaşmak için bağlantıların nasıl değiştirileceği, öğrenme algoritması tarafından belirlenir. Kullanılan bir öğrenme kuralına göre, hatayı sıfıra indirecek şekilde ağın ağırlıkları değiştirilir. Yapay sinir ağları yapılarına ve öğrenme algoritmalarına göre sınıflandırılırlar.

2.4.1. Yapay sinir ağlarının yapılarına göre sınıflandırılması

Yapay sinir ağları, yapılarına göre, ileri beslemeli (feedforward) ve geri beslemeli (feedback) ağlar olmak üzere iki şekilde sınıflandırılırlar. Bir çok yapay sinir ağı bulunmakla birlikte, en çok kullanılan sinir ağı yapısı, İleri Beslemeli Geri Yayılımlı yapay sinir ağı olarak bilinendir (Erdem ve Uzun 2005).

2.4.1.a. İleri beslemeli ağlar

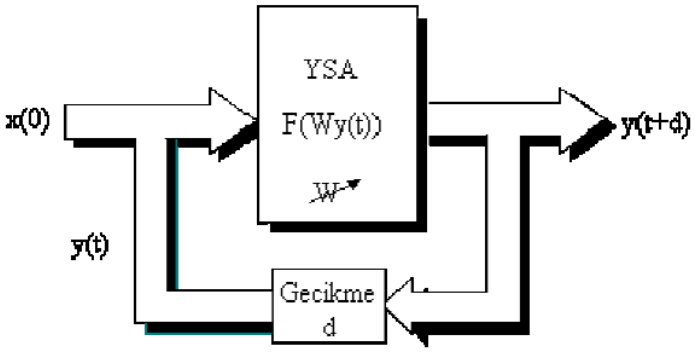
İleri beslemeli bir ağda işlemci elemanlar (İE), genellikle katmanlara ayrılmışlardır. Her bir katmandaki hücreler sadece bir önceki katmanın hücrelerince beslenir. İleri beslemeli YSA'da, hücreler katmanlar şeklinde düzenlenir ve bir katmandaki hücrelerin çıkışları bir sonraki katmana ağırlıklar üzerinden giriş olarak verilir. Giriş katmanı dış ortamdan aldığı bilgileri hiçbir değişikliğe uğratmadan orta (gizli) katmandaki hücrelere iletir. Bilgi, orta ve çıkış katmanında işleyerek ağ çıkışı belirlenir (Kaya ve Engin 2005). İşaretler, giriş katmanından çıkış katmanına doğru tek yönlü bağlantılarla iletilir. İE'ler bir katmandan diğer bir katmana bağlantı kurarlarken, aynı katman içerisinde bağlantıları bulunmaz. Şekil 2.10'da ileri beslemeli ağ için blok diyagram gösterilmiştir. İleri beslemeli ağlara örnek olarak Çok Katmanlı Algılayıcı (Multi Layer Perceptron – MLP) ve Doğrusal Vektör Parçalama (Linear Vector Quantization – LVQ) ağları verilebilir.



Şekil 2.10. İleri Beslemeli Ağ İçin Blok Diyagram

2.4.1.b. Geri beslemeli ağlar

Şekil 2.11'de bir geri beslemeli ağ görülmektedir. Bu çeşit sinir ağlarının dinamik hafızaları vardır ve bir andaki çıkış, hem o andaki hem de önceki girişleri yansıtır. Bundan dolayı, özellikle tahmin uygulamalarında kullanılırlar. Bu ağlar çeşitli tipteki problemlerin tahmininde oldukça başarı sağlamışlardır. Bu ağlara örnek olarak Hopfield, Düzenleyici Harita (Self Organizing Map – SOM), Elman ve Jordan ağları verilebilir.



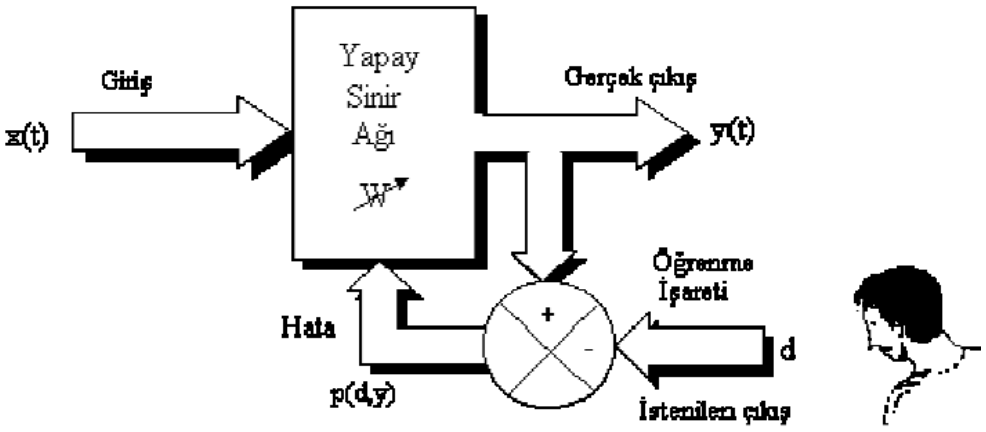
Şekil 2.11. Geri Beslemeli Ağ İçin Blok Diyagram

2.5. Yapay Sinir Ağlarının Öğrenme Algoritmalarına Göre Sınıflandırılması

Genel olarak üç öğrenme metodundan ve bunların uygulandığı değişik öğrenme kurallarından söz edilebilir. Bu öğrenme kuralları şu şekilde açıklanmaktadır:

2.5.1. Danışmanlı öğrenme (Supervised learning)

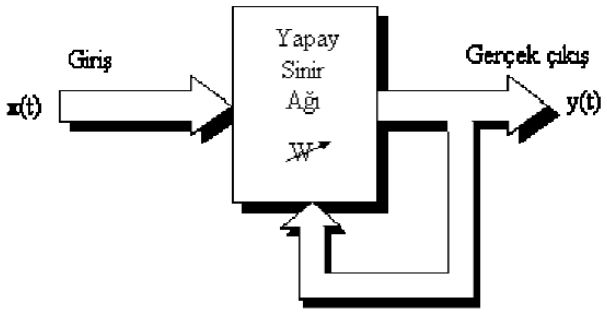
Bu tip öğrenmede, yapay sinir ağlarına örnek olarak bir doğru çıkış verilir. İstenilen ve gerçek çıktı arasındaki farka (hataya) göre İE'ler arası bağlantıların ağırlıkları en uygun çıkışı elde etmek için sonradan düzenlenebilir. Bu sebeple danışmanlı öğrenme algoritmasının bir "öğretmene" veya "danışmana" ihtiyacı vardır. Widrow-Hoff tarafından geliştirilen delta kuralı ve Rumelhart ve McClelland tarafından geliştirilen genelleştirilmiş delta kuralı veya geri yayılım (back propagation) algoritması, danışmanlı öğrenme algoritmalarına örnek olarak verilebilir. Danışmanlı öğrenmenin şekilsel gösterimi Şekil 2.12'de görüldüğü gibidir.



Şekil 2.12. Danışmanlı Öğrenme Yapısı

2.5.2. Danışmansız öğrenme (Unsupervised learning)

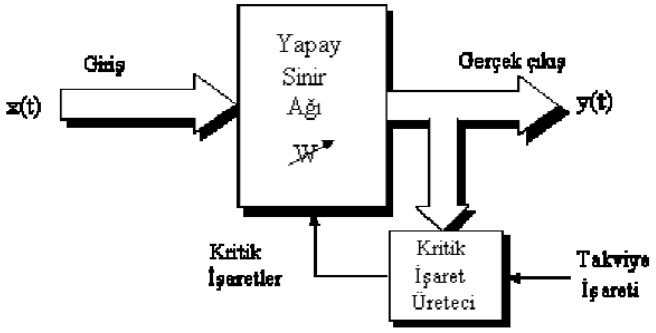
Girişe verilen örnekten elde edilen çıkış bilgisine göre ağ, sınıflandırma kurallarını kendi kendine geliştirmektedir. Bu öğrenme algoritmalarında, istenilen çıkış değerinin bilinmesine gerek yoktur. Öğrenme süresince sadece giriş bilgileri verilir. Ağ daha sonra bağlantı ağırlıklarını, aynı özellikleri gösteren desenler (patterns) oluşturmak üzere ayarlar. Grossberg tarafından geliştirilen Adaptif Rezonans Teorisi (Adaptive Resonance Theory – ART) veya Kohonen tarafından geliştirilen SOM öğrenme kuralı, danışmansız öğrenmeye örnek olarak verilebilir. Danışmansız öğrenme yapısı Şekil 2.13'de olduğu gibidir.



Şekil 2.13. Danışmansız Öğrenme Yapısı

2.5.3. Takviyeli öğrenme (Reinforcement learning)

Bu öğrenme kuralı danışmanlı öğrenmeye yakın bir metottur. Denetimsiz öğrenme algoritması, istenilen çıkışın bilinmesine gerek duymaz. Hedef çıktıyı vermek için bir "öğretmen" yerine, burada yapay sinir ağına bir çıkış verilmemekte fakat elde edilen çıkışın, verilen girişe karşılık iyiliğini değerlendiren bir kriter kullanılmaktadır. Optimizasyon problemlerini çözmek için Hinton ve Sejnowski'nin geliştirdiği Boltzmann kuralı veya genetik algoritmalar, takviyeli öğrenmeye örnek olarak verilebilirler. Takviyeli öğrenme yapısı Şekil 2.14'de olduğu gibidir.



Şekil 2.14. Takviyeli Öğrenme Yapısı

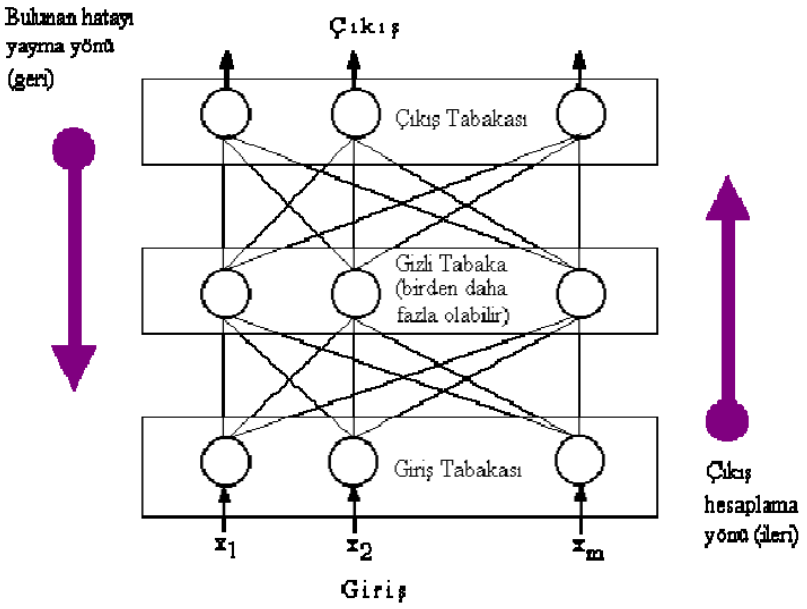
2.6. Çok Katmanlı Algılayıcılar ve Öğrenme Algoritmaları

Çok katmanlı algılayıcılar, birçok öğretme algoritması kullanılarak eğitilebilirler.

2.6.1. Çok katmanlı algılayıcılar (MLP)

Çok katmanlı bir algılayıcı (perceptron) sinir ağı modeli, Şekil 2.15'de gösterilmiştir. Bu ağ modeli özellikle mühendislik uygulamalarında en çok kullanılan sinir ağı modeli olmuştur. Bir MLP modeli, bir giriş, bir veya daha fazla ara ve bir de çıkış katmanından oluşur. Bir katmandaki bütün işlem elemanları bir üst katmandaki bütün işlem elemanlarına bağlıdır. Bilgi akışı ileri doğru olup geri besleme yoktur.

Bunun için ileri beslemeli sinir ağı modeli olarak adlandırılır. Giriş katmanında herhangi bir bilgi işleme yapılmaz. Buradaki işlem elemanı sayısı, tamamen uygulanan problemin giriş sayısına bağlıdır. Ara katman sayısı ve ara katmanlardaki işlem elemanı sayısı ise, deneme-yanılma yolu ile bulunur. Çıkış katmanındaki eleman sayısı ise yine uygulanan probleme dayanılarak belirlenir.



Şekil 2.15. Geri Yayılım Çok Katmanlı Algılayıcı Yapısı

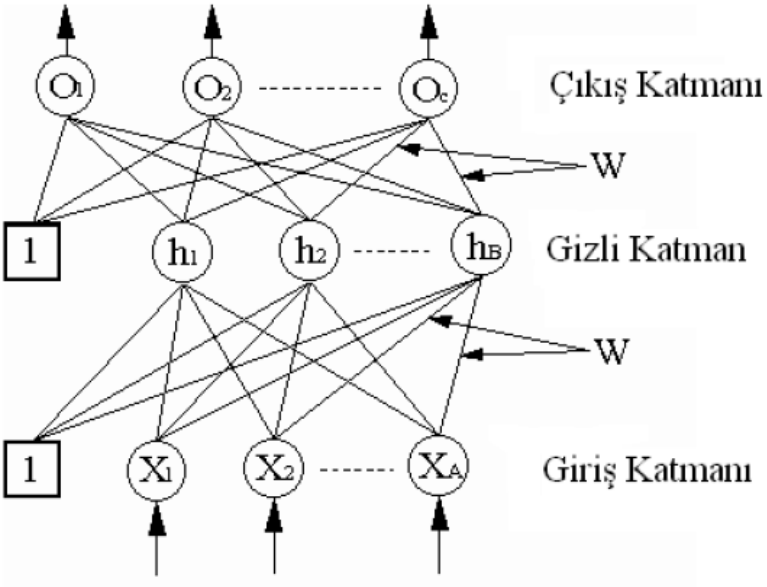
Doğrusal olmayan problemlerin çözümlenmesinde giriş katmanı ile çıkış katmanı arasında ilave katman(lar) yer aldığı için ağ mimarisi çok katmanlı olmaktadır. Sinir ağlarının mimarisinde gizli katman sayısının belirlenmesi oldukça önemlidir. Giriş ve çıkış katmanlarının oluşturulmasından farklı olarak ağ mimarisi gerçekleştirilirken gizli katman sayısı hakkında ön bilgi olmadan işleme başlanır. Birkaç gizli nöronu olan ağ karmaşık örüntüleri ayırt edememektedir çünkü sadece doğrusal kestirim yapabilmektedir. Bunun yanı sıra, gizli nöron sayısının fazla olması ağın genelleştirme yapmasını engellemektedir. Gizli katman sayısının artırılması durumunda ağın eğitimi için geçen süre artmaktadır (Güler ve Übeyli 2006).

Aynı zamanda düğüm noktası ve aralarındaki bağlantı sayısı arttıkça YSA'nın öğrenme kapasitesi artmakta, fakat eğitim zamanı da artmaktadır (Atik 2006). MLP ağlarında, ağa bir örnek gösterilir ve örnek neticesinde nasıl bir sonuç üreteceği de bildirilir (danışmanlı öğrenme). Örnekler giriş katmanına uygulanır, ara katmanlarda işlenir ve çıkış katmanından da çıkışlar elde edilir. Kullanılan eğitme algoritmasına göre, ağın çıkışı ile arzu edilen çıkış arasındaki hata geriye doğru yayılarak, hata minimuma düşünceye kadar ağın ağırlıkları değiştirilir.

İleri beslemeli ağlar, en genel anlamıyla giriş uzayıyla çıkış uzayı arasında statik haritalama yapar. Bir andaki çıkış, sadece o andaki girişin bir fonksiyonudur.

2.6.2. Geri yayılım algoritması

Giriş, çıkış ve en az bir tane gizli katman olmak üzere üç katmandan oluşmaktadır. Her katmandaki düğümler, kendisinden bir önceki ve sonraki katmandaki tüm düğümlere bağlıdırlar. Geri yayılım algoritmasına aynı zamanda Çok Katmanlı Algılayıcı (ÇKA) da denilmektedir. Geri yayılım algoritması yapısı Şekil 2.16'da görüldüğü gibidir.



Şekil 2.16. Geri Yayılım Algoritması Mimarisi

Giriş katmanı dış dünyadan gelen girdileri (X_1, X_2,…, X_n) alarak ara katmana gönderir. Bu katmanda bilgi işleme olmaz. Gelen her bilgi geldiği gibi bir sonraki katmana gider. Birden fazla girdi gelebilir. Her proses elemanının sadece bir tane girdisi ve bir tane çıktısı vardır. Bu çıktı bir sonraki katmanda bulunan bütün proses elemanlarına gönderilir. Yani giriş katmanındaki her proses elemanı bir sonraki katmanda bulunan proses elemanlarının hepsine bağlıdır.

Gizli katman giriş katmanından gelen bilgileri işleyerek bir sonraki katmana gönderir. Bir geri yayılım algoritması ağında birden fazla proses elemanı olabilir. Ara katmandaki her proses elemanı bir sonraki katmandaki bütün proses elemanlarına bağlıdır.

Çıktı katmanı ara katmandan gelen bilgileri işleyerek ağa giriş katmanından verilen girdilere karşılık ağın ürettiği çıktıları (O_1, O_2, …, O_n) belirleyerek dış dünyaya gönderir. Her proses elemanı bir önceki katmanda bulunan bütün proses elemanlarına bağlıdır. Her proses elemanının sadece bir tane çıktısı vardır. ÇKA ağında bilgiler girdi katmanından ağa sunulur ve ara katmanlardan geçerek çıktı katmanına gider ve ağa sunulan girdilere karşılık ağın cevabı dış dünyaya iletilir (Öztemel 2003).

ÇKA ağı veyahut geri yayılım algoritmasında en çok kullanılan öğreticili öğrenme algoritmasıdır (öğretmenli öğrenme stratejisi) (Hamzaçelebi ve Kutay 2004). ÇKA ağının öğrenme kuralı en küçük kareler yöntemine dayalı Delta Öğrenme Kuralının genelleştirilmiş halidir. O nedenle öğrenme kuralına Genelleştirilmiş Delta Kuralı da denmektedir. Ağın öğrenebilmesi için eğitim seti adı verilen ve örneklerden oluşan bir sete ihtiyaç vardır. Bu set içinde her bir örnek için ağın hem girdiler hem de o girdiler için ağın üretmesi gereken çıktılar belirlenmiştir. Genelleştirilmiş “Delta Kuralı” iki safhadan oluşur.

1- Safha-ileri doğru hesaplama: Ağın çıktısını hesaplama safhasıdır.

2-Safha-geriye doğru hesaplama: Ağırlıkları değiştirme safhasıdır.

Proses elemanlarını birbirine bağlayan ağırlık değerlerinin ve eşik değer ünitesinin ağırlıklarının başlangıç değerlerinin atanması yapılır. Başlangıçta genellikle rastgele değerler atanır. Genel olarak ağırlıklar belirli aralıklarla atanmaktadır. Bu aralık eğer büyük tutulursa ağın yerel çözümler arasında sürekli dolaştığı küçük olması durumunda ise öğrenmenin geç gerçekleştiği görülmüştür.

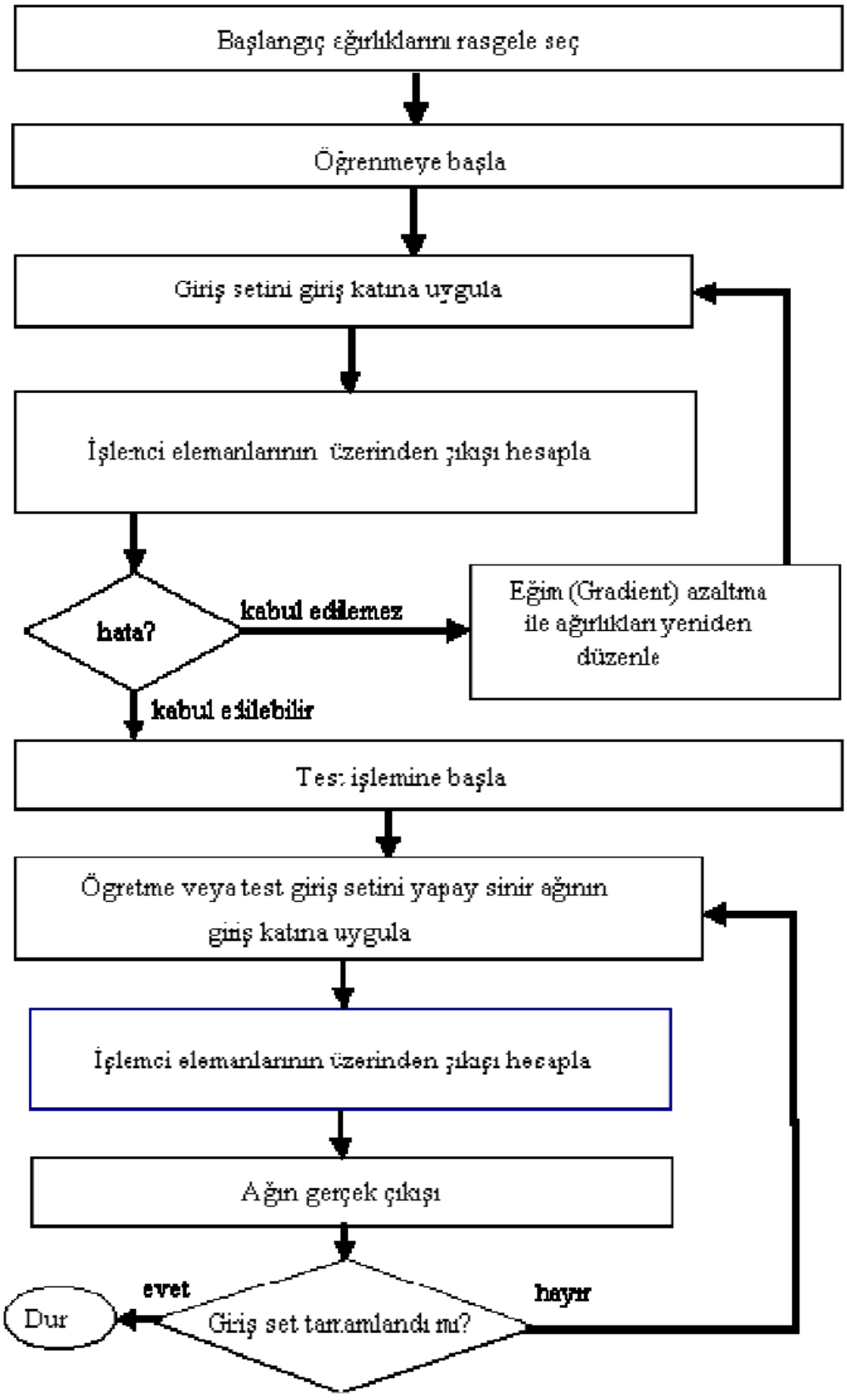
Başlangıç değeri kadar öğrenme ve momentum katsayılarının belirlenmesi de ağın öğrenme performansı ile yakından ilgilidir. Öğrenme katsayısı ağırlıkların değişim miktarını belirlemektedir. Eğer büyük değerler seçilirse o zaman yerel çözümler arasında ağın dolaşması ve osilasyon yaşaması söz konusu olmaktadır. Küçük değerler seçilmesi ise öğrenme zamanını artırmaktadır. Tecrübeler genellikle 0.2-0.4 arasındaki değerlerin kullanıldığını göstermektedir. Fakat bu tamamen ilgili probleme bağlıdır. Bu değerler iyidir demek de doğru olmaz. Bazı uygulamalarda öğrenme katsayısının 0.6 değerini aldığı zaman en başarılı sonuçları verdiği görülmektedir.

Benzer şekilde momentum katsayısı da öğrenmenin performansını etkiler. Bu özellikle yerel çözümlere takılan ağların bir sıçrama ile daha iyi sonuçlar bulmasını sağlamak amacı ile önerilmiştir. Bu değerin küçük olması yerel çözümlerden kurtulmayı zorlaştırabilir. Çok büyük değerler ise tek bir çözüme ulaşmada sorunlar yaşanabilir (Öztemel 2003). Momentum değeri 0 ile 1 arasındadır (Bayındır ve Sesveren 2008). Tecrübeler bu değerin 0.6-0.8 arasında seçilmesinin uygun olacağını göstermektedir. Fakat bu da kesin denilemez. Problemin niteliğine göre kullanıcının belirlenmesinde fayda vardır. Daha küçük değerler ile başarılı sonuçların alındığını gösteren örnekleri görmek de mümkündür.

ÇKA modelinde ağın eğitilmesi kadar gereğinden fazla eğitilmemesi de önemlidir. Çünkü eğitilmek istenen bir ağ problem uzayına çözüm üretecek ağırlıkları bulduktan sonra eğitime devam edilirse bu ağın ağırlıklarında daha fazla değişikliklere neden olur. Böylece en iyi çözüm üreten bir ağ tekrar daha performansı düşük ağlara veya öğrenemeyen ağlara dönüşebilir. O nedenle ağın eğitiminin ne zaman durdurulması gerektiği konusunda da karar vermek gerekmektedir. Bu da ağın başarısını ve

performansını etkilemektedir. Pratikte, genel olarak iki türlü durdurma kriteri kullanılmaktadır. Hatanın belirli değerin altına düşmesi halinde eğitimi durdurma, ağın belirli bir iterasyon sayısını tamamlaması sonucu eğitimi durdurma (Öztemel 2003).

Geri yayılım algoritmasının çalışma şekli Şekil 2.17'de görüldüğü gibidir.



Şekil 2.17. Çok Katmanlı Algılayıcının Geri Yayılım Akış Seması

Geri yayılım algoritması çok yaygın olarak kullanılmasına rağmen yöntemin uygulamadaki başarısı ve güvenilirliği için bazı ayrıntılara dikkat etmek gerekmektedir.

Ağa uygulanan ilk değerlere ve eğitim devir sayısına bağlı olarak eğitim süreci yüzlerce ya da milyonlarca devir sürebilir. Bu konuda kesin bir sayı yoktur. Burada kullanıcının tecrübesi, ağı uygun tasarlaması ve parametreleri uygun seçmesi belirleyici olur. Bir GYA ağının yapısı, eğitim süresi boyunca N parametreli bir ağ için N+1 boyutlu bir uzayda, N değişkenli bir yüzey üzerinde gezen bir noktanın, hatayı en aza indiren noktayı aramasını gerektirmektedir. Ağın içinde yapılan türev işlemlerinin sonucuna göre bazı durumlar için eğitim işlemi çok uzun sürebilmektedir. Bu durumu engellemek için sisteme η öğrenme katsayısı, μ momentum terimi ve her bir hücre için bir eşik değeri (bias) ilave edilir.

$$w_{pq,k}(N+1) = w_{pq,k}(N) - \eta_{p,q}\delta_{pq,k}\Phi_{p,j}(\mathrm{I}) + \mu\Delta w_{pq,k}(N)$$

Yukarıdaki denklemde;

w : ağırlık değeri,
p, q : hücre numarası,
k : katman sayısı,
N : devir sayacı,
H : öğrenme katsayısı,
d : geriye yayılacak hata terimi,
μ : momentum,
Φ(I) : ağın bulduğu sonuçtur.

Ağa eklenen eşik (bias) elemanı ağın öğrenme hızını genelde olumlu yönde etkiler.

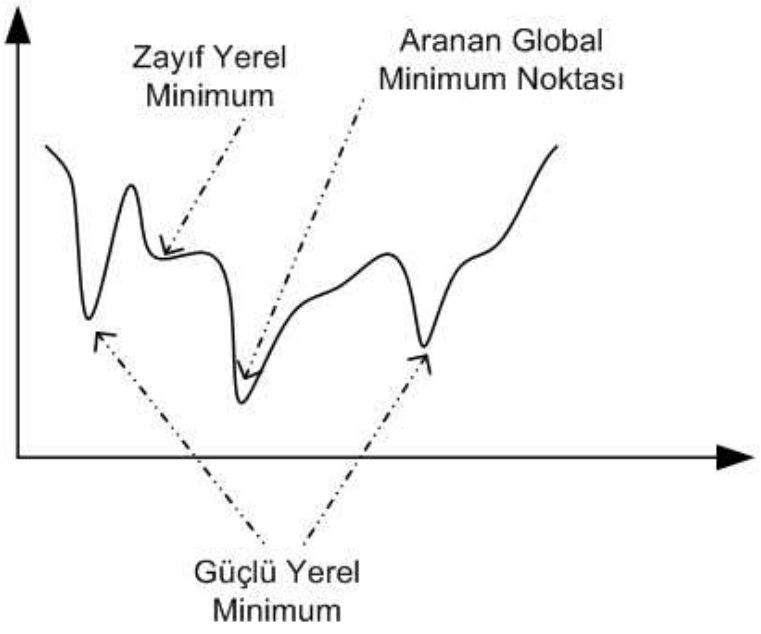
Momentum değeri ise eğitim sürecinin yönünün korunmasını sağlar. Bunun için ağırlıkların güncellenmesi sırasında önceki ağırlık değişimiyle orantılı olarak bir momentum değeri ilave edilir.

Öğrenme katsayısı pozitif bir değer olmak zorundadır. Eğer 2'den büyük seçilirse ağın kararsızlığına neden olur. 1'den büyük seçildiğinde ise ağın çözüme ulaşması yerine sabit bir aralıkta salınım yapmasına neden olabilir. Öğrenme katsayısı için uygun değerler (0,1) aralığındadır. Bu aralıkta seçilecek katsayının büyüklüğüyle öğrenme adım aralığı doğru orantılıdır.

Sigmoid fonksiyonu kullanan bir ağda artan ağırlık değerleri sigmoid fonksiyonunun türevinin çok küçük değerli bölgelerde işlem yapmasına neden olur.

Geriye yayılan hata terimi türevle orantılı olduğundan bu durumda yeterli eğitim gerçekleştirilemez.

Ağın Yerel Minimum Noktasına Takılması



Şekil 2.18. Hata Değerlerinin Seyri

Geriye yayılım algoritmasının en büyük zaafı, türev işlemleri sırasında yerel minimum noktasında takılması ve bu noktayı en iyi çözüm zannetmesidir. Bu durum "ağın ezberlemesi" olarak da adlandırılır. Toplam hata değerinin kabul edilebilir olduğu durumlarda bulunan noktanın yerel ya da bölgesel minimum noktası olması herhangi bir sorun yaratmayabilir ancak henüz toplam hata değeri kabul edilemez bir noktada iken ağın eğitimi kesilirse ağın üreteceği sonuçlar hatalı olacaktır.

2.6.2.a. Başlangıç değerlerinin belirlenmesi

Proses elemanlarını birbirine bağlayan ağırlık değerlerinin ve eşik değer ünitesinin ağırlıklarının başlangıç değerlerinin atanması yapılır. Başlangıçta genellikle rastgele değerler atanır. Genel olarak ağırlıklar belirli aralıklarla atanmaktadır.

Öğrenme katsayısı ağırlıkların değişim miktarını belirlemektedir. Eğer büyük değerler seçilirse o zaman yerel çözümler arasında ağın dolaşması ve osilasyon yaşaması söz konusu olmaktadır. Küçük değerler seçilmesi ise öğrenme zamanını artırmaktadır.

Benzer şekilde momentum katsayısı da öğrenmenin performansını etkiler. Bu özellikle yerel çözümlere takılan ağların bir sıçrama ile daha iyi sonuçlar bulmasını sağlamak amacı ile önerilmiştir.

2.6.2.b. Ağın boyutları ve yapısı

Çok katmanlı bir yapıya sahip olan geriye yayılımlı yapay sinir ağlarında sadece giriş, ara ve çıkış katmanından oluşan toplam 3 katmanlık bir ağ her zaman yeterli olmayabilir. Katman sayısı ve her katmandaki hücre sayısı yine kullanıcın vereceği bir karardır. Giriş ve çıkış katmanlarındaki düğüm sayısı problemin yapısıyla ilgilidir. Giriş verilerinin eleman sayısı (kaç değişik girdinin olduğu bireysel, şirket, kartlı, kartsız gibi) giriş katmanının düğüm sayısını ve beklenen sonuç da çıkış katmanının düğüm sayısını belirler. Gizli katmandaki düğüm sayısı ise kullanıcının tecrübesine bağlıdır. Fazladan konulacak her bir düğüm ağın yavaşlamasına neden olur.

2.6.2.c. Eğitim verilerinin seçilmesi ve ağ için uyarlanması

Elde bulunan tüm verileri ağın eğitiminde kullanmak doğru bir yöntem değildir. Genellikle eldeki verilerin %70'i eğitim, %20'si onaylama ve %10'u test için kullanılır.

Eğitimde kullanılacak olan verilerin problem uzayını doğru temsil etmesi de çok önemlidir. Ağı, çözülmeye çalışılan problemle ilgisi göreceli olarak az verilerle eğitmek ağın başarı oranını ciddi şekilde düşürecektir (Yıldız 2006).

2.7. Yapay Sinir Ağlarında Öğrenme Kuralları

Yapay sinir ağlarının mimarisi kadar ağın eğitilmesinde yani ağın öğrenmesinde kullanılacak yöntem de çok önemlidir. Genel olarak ifade etmek gerekirse YSA'lar da bir bebeğin beyninin öğrenmesi gibi deneme yanılmayla, hata yapa yapa öğrenir. Eğitim sırasındaki amaç bulunması gereken doğru sonuçlara en yakın çıktıyı üretebilmektir. Bu sebeple ağ verilen girdilere göre kendi mimarisine de uygun olarak işlem yaptıktan sonra bir çıktı üretir. Çıktı ile hedef değerler arasındaki fark hatadır. Ağ bu hatayı kabul edilebilir sınırlara arasına indirebilmek için işlemi tekrarlar. Eğitim setinin ağ içinde bir kez işlemden geçirilmesine devir (epoch) denir. Devir sayısının çok olması ağın öğrenme sürecinde önemli bir etkendir.

Fakat devir sayısının yüksek seçilmesi de performansı düşüren bir etkendir. Bu durumda ağın mimarisi, aktivasyon fonksiyonu, öğrenme yöntemi ve devir sayısı seçilirken optimizasyonun iyi bir şekilde yapılması gerekir.

En bilinen ve en yaygın olarak kullanılan öğrenme kuralları şunlardır:

2.7.1. Hebb kuralı (1949)

Diğer öğrenme kurallarının da temelini oluşturan bu kurala göre, bir sinir hücresi diğer bir hücreden bilgi alırsa ve eğer her ikisi de matematiksel olarak aynı işareti taşıyorsa yani aktif ise bu iki hücre arasındaki bağlantı kuvvetlendirilmelidir. Tersi durumda ise zayıflatılmalıdır.

2.7.2. Hopfield kuralı (1982)

Bu kural Hebb kuralına benzemektedir. Diğerinden farklı olarak YSA elemanlarının bağlantılarının ne kadar kuvvetlendirilmesi ya da zayıflatılması gerektiğini de belirler. Eğer beklenen çıktıların ve girdilerin her ikisi de aktif/pasif ise öğrenme katsayısı kadar ağırlık değerlerini de kuvvetlendirir/zayıflatır.

Ağırlıkların kuvvetlendirilmesi ya da zayıflatılması öğrenme katsayısı yardımıyla gerçekleştirilir. Bu katsayı genellikle 0 ile 1 arasında kullanıcı tarafından belirlenen sabit bir pozitif değerdir.

2.7.3. Delta kuralı

Bu kurala göre hedef çıktı ile elde edilen çıktı arasındaki farkı azaltmak için YSA elemanlarının bağlantılarının ağırlık değerleri sürekli yeniden hesaplanır. Amaç hedef çıktı ile elde edilen çıktı arasındaki hata karelerinin ortalamasını en aza indirebilmektir. Hatalar en son katmandan geriye doğru ardışık iki katman arasındaki bağlantı ağırlıklarına dağıtılır. Bu isleme hatanın geriye dağıtılması anlamında geri besleme işlemi denir.

2.7.4. Kohonen kuralı (1998)

Bu kural biyolojik sinir hücrelerinin öğrenme kurallarından esinlenerek oluşturulmuştur. Bu kuralda sinir hücreleri ağırlıkları değiştirmek için birbirleri ile yarışırlar. En büyük çıktıyı üreten hücre kazanan hücre olur ve bağlantı ağırlıklarını değiştirir. Kazan hücre yakınındaki hücrelere göre daha kuvvetli hale gelmektedir.

Bu kuralda bir hedef değerler dizisi olmasına gerek yoktur. Bu nedenle kendi kendine yani öğretmensiz olarak eğitimini tamamlar.

2.8. Yapay Sinir Ağlarında Öğrenme Stratejileri

2.8.1. Öğretmenli eğitim

Öğretmenli eğitimde, verilen giriş değerlerine karşılık gelen hedef çıktı değerleri vardır. Ağın görevi verilen değerlere göre hedef çıktıyı üretebilmektir. Ağın çıktıları sanki bir öğretmen varmışçasına hedef değerlerle kıyaslanır ve kabul edilebilir değerler arasında olup olmadığına göre eğitime devam edilir. Hedef değerler ile elde edilen değerler arasındaki farkın yani hatanın karelerinin ortalaması en küçük olacak şekilde ağırlıklar sürekli olarak güncellenir ve işlemlere devam edilir.

Bu yöntemde ağa hedef çıktılar ile elde edilen verilerin arasındaki farkın verilebileceği gibi ağa sadece sonucun doğru mu yoksa yanlış mı olduğu da söylenebilir. İkinci yöntem destekleyicili öğrenme olarak da adlandırılır. Delta Kuralı, Genelleştirilmiş Delta Kuralı, Rastsal Öğrenme Kuralı, Takviyeli Öğrenme Kuralı öğretmenli eğitimin kurallarından bazılarıdır.

2.8.2. Öğretmensiz eğitim

Bu tip eğitimde hedef çıktı değerleri yoktur. Ağa sadece giriş değerleri verilir. Örneklerdeki parametreler arasındaki ilişkileri ağın kendi kendine öğrenmesi beklenir. Daha çok sınıflandırma problemlerinde kullanılan bir eğitim yöntemidir.

Sınıflandırma işleminde hedef mümkün olduğunca farklı sayıda sınıflandırma yapabilmektir. Daha sonra kullanıcının elde edilen sınıfların ne anlama geldiğini kendisinin yapması gerekir.

Signal Hebbian Öğrenme Kuralı ve Diferansiyel Hebbian Öğrenme Kuralı öğretmensiz eğitim kurallarına örnek olarak verilebilir.

2.8.3. Karma eğitim

Yukarıda bahsedilen yöntemlerin birkaçını birlikte kullanarak yapılan eğitim işlemidir. Yani ağ, kısmen öğretmenli kısmen de öğretmensiz olarak öğrenme işlemini gerçekleştirir. Radyal tabanlı YSA'lar ve olasılık tabanlı ağlar bunlara örnek verilebilir. Öğrenme algoritmaları ve kullanım alanları Çizelge 2.1.'de görüldüğü gibidir.

Çizelge 2.1. Öğrenme Algoritmaları ve Kullanım Alanları Tablosu

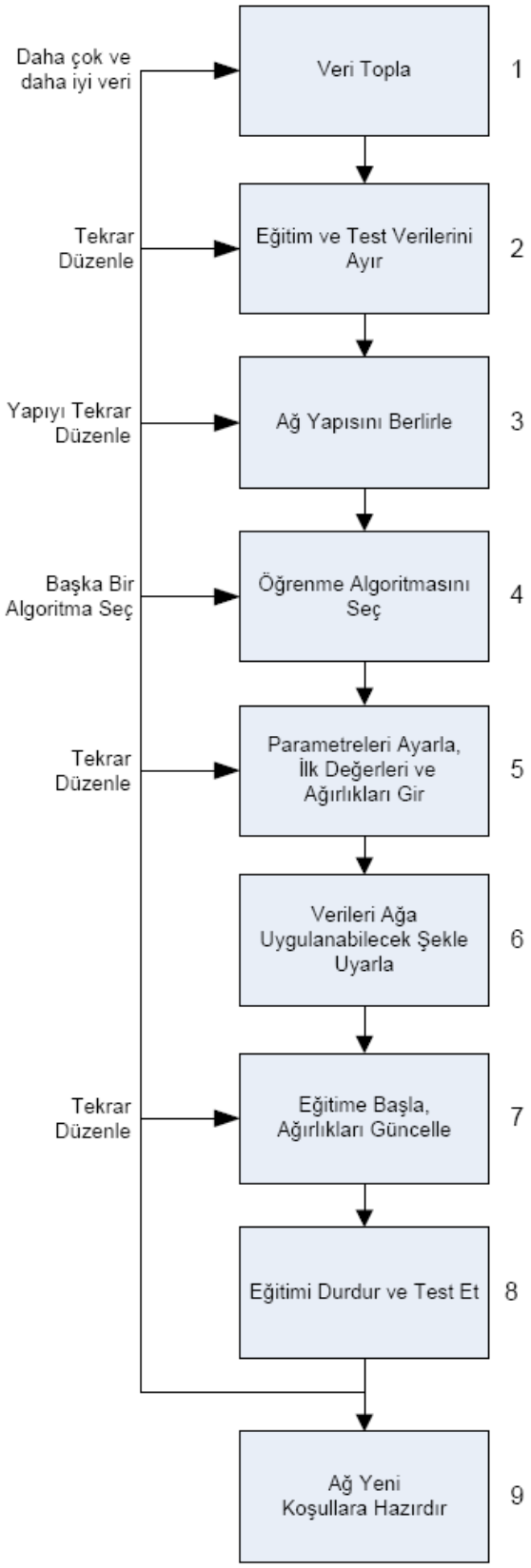
Öğrenme Yöntemi	Öğrenme Kuralı	Ağın Mimarisi	Öğrenme Algoritması	Kullanıldığı Yerler
Öğretmenli	Hata Düzeltme	Tek ya da çok katmanlı	Perseptron Geriye Yayılım Adaline ve Madaline[88]	Örüntü tanıma Tahmin Kontrol
	Boltzmann	Geri Dönüşlü	Boltzmann öğrenmesi	Örüntü sınıflandırma
	Hebbian	Çok katmanlı ileri beslemeli	Lineer diskriminant	Veri analizi Örüntü sınıflandırma
	Rekabetli	Rekabetli	Vektör kuantalama öğrenmesi	Veri sıkıştırma Sınıflandırma
		ART ağı	ART Haritalaması	Örüntü sınıflandırma Sınıflandırma
Öğretmensiz	Hata Düzeltme	Çok katmanlı ileri beslemeli	Sammon projeksiyonu	Veri analizi
	Hebbian	İleri beslemeli rekabetçi	Principal Component Analizi	Veri analizi Veri sıkıştırma
		Hopfield Ağı	Associative Memory Öğrenmesi	Associative Memory
	Rekabetli	Rekabetli	Vektör Kuantalaması	Sınıflandırma Veri sıkıştırma
		Kohonen SOM	Kohonen SOM	Sınıflandırma Veri analizi
		ART ağı	ART1, ART2	Sınıflandırma
Karma	Hata Düzeltme ve Rekabetli	Radyal Tabanlı	Radyal Tabanlı Öğrenme	Sınıflandırma Örüntü tanıma Fonksiyon yaklaşımı Tahmin Kontrol

2.9. Diğer Yapay Sinir Ağları

1. LVQ (Learning Vector Quantization)
2. Hopfield ağı
3. Elman ve Jordan Ağları
4. Kohonen Ağı
5. ART (Adaptive Resonance Theory) ağı

2.10. Bir YSA Modellemesinde Dikkat Edilecek Hususlar

Bir problemi yapay sinir ağı kullanarak çözmek istediğimizde aşağıdaki süreci uygulamamız gerekir. Yapay sinir ağının çalışma prensibi Şekil 2.19'da görüldüğü gibidir.



Şekil 2.19. YSA Modellemesi Akış Seması

1. Ağın eğitiminde ve testinde kullanılacak olan veriler toplanır. Kaç değişkenin girdi ve kaçının çıktı olacağına karar verilmelidir. Böylece giriş ve çıkış katmanında kaçar hücre olacağı da belirlenmiş olur.

2. Toplanan veriler problemin türüne verilere de bağlı olarak eğitim ve test için ayrıştırılır. Belirli bir oran olmamasına rağmen veriler genellikle %70, %20 ve %10 luk paketlere ayrılır. Eğer veri dizisinde sıra önemli değilse ayrıştırma rastgele yapılırsa yani örnekler veri setinin her yerinden alınırsa eğitim daha başarılı olur.

3. Problemin türüne göre ağın yapısına karar verilir.

4. Problemin türüne göre öğrenme yöntemi seçilir.

5. Ağa uygulanacak ilk değerler verilir.

6. Veriler ağa verilecek şekilde uyarlanır ve ağın giriş katmanına uygulanır.

7. Eğitime başlanır, hatayı en aza indirmek için ağırlıklar sürekli gözden geçirilir.

8. Hata kabul edilebilir sınırlar arasına girdiğinde eğitim durdurulur ve ağ test edilir. Ayrıştırılan verinin ilk paketi ile eğitim gerçekleştirildikten sonra elde edilen ağırlıklar ile ikinci paket veri ağa uygulanır. Eğer ikinci eğitimden (onaylama - validation) sonra ağırlıklardaki değişimler ihmal edilebilir düzeyde ise ağ test edilmeye hazır demektir.

Bu asamadan sonra, sonuçlar tatmin edici ise ağımız kullanıma hazır demektir.

Eğer testlerde kabul edilebilir sonuçları elde edemediysek:

1. Eğitim için daha fazla ve daha uygun veriler toplanarak ağa uygulanabilir,

2. Eğitim ve test için ayrılan veriler yeniden düzenlenebilir,

3. Ağın yapısı değiştirilebilir ya da başka bir model YSA kullanılabilir,

4. Öğrenme algoritması değiştirilebilir,

5. Başlangıç değerleri yenilenerek ağ yeniden eğitilebilir (Yıldız 2006).

2.11. Yapay Sinir Ağlarının Avantajları

YSA'ları, makine öğrenmesi gerçekleştirebilirler. Yapay sinir ağlarının temel işlevi bilgisayarın öğrenmesini sağlamaktır. Olayları öğrenerek benzer olaylar karsısında mantıklı kararlar verebilirler.

Bilgi işleme yöntemleri geleneksel programlamadan farklıdır. Bu nedenle geleneksel programlamanın getirdiği bir çok olumsuzluk ortadan kaldırılabilir.

Bilgiler ağın tamamında saklanır. Geleneksel programlamada olduğu gibi, bilgiler veri tabanları ya da dosyalarda belli bir düzende tutulmaz, ağın tamamına yayılarak değerler ile ölçülen ağ bağlantılarında saklanmaktadır. Hücrelerden bazılarının işlevini yitirmesi, anlamlı bilginin kaybolmasına neden olmaz.

Örnekleri kullanarak öğrenirler. YSA'nın öğrenebilmesi için örneklerin belirlenmesi, bu örneklerin ağa gösterilerek istenilen çıktılara göre ağın eğitilmesi gerekmektedir. Ağın başarısı, seçilen örnekler ile doğru orantılıdır, ağa olay bütün yönleri ile gösterilemezse ağ yanlış çıktılar üretebilir.

Daha önce görülmemiş örnekler hakkında bilgi üretebilirler. YSA'ları eğitimleri sırasında kendilerine verilen örneklerden genellemeler çıkarırlar ve bu genellemeler ile yeni örnekler hakkında bilgi üretebilirler.

Algılamaya yönelik olaylarda kullanılabilirler. YSA'larının en başarılı oldukları alanlar, algılamaya yönelik uygulama alanlarıdır. Bu alanlarda başarıları kanıtlanmıştır.

Örüntü (pattern) ilişkilendirme ve sınıflandırma yapabilirler. YSA'ları kendilerine örnekler halinde verilen örüntüleri kendisi veya diğerleri ile ilişkilendirebilir. Ayrıca kendisine verilen örneklerin kümelenmesi ile, bir sonraki verinin hangi kümeye dahil olacağına karar verilmesi konusunda kullanılabilirler.

Örüntü tamamlama yapabilirler. Ağa eksik bilgileri içeren örüntüler verildiğinde eksik bilgilerin tamamlanması konusunda başarılıdırlar.

Kendi kendine öğrenebilme ve organize etme yetenekleri vardır. YSA'ları online olarak öğrenebilirler ve kendi kendilerini eğitebilirler.

Eksik bilgi ile çalışabilmektedirler. Geleneksel sistemlerin aksine YSA'ları eğitildikten sonra veriler eksik bilgi içerse dahi, çıktı üretebilirler. Bu durum bir performans kaybı yaratmaz, performans kaybı eksik bilginin önemine bağlıdır.

Burada bilgilerin önem dereceleri eğitim sırasında öğrenilir.

Hata toleransına sahiptirler. YSA'larının, eksik bilgilerle çalışabilmeleri ve bazı hücreleri bozulsa dahi çalışabilmeleri, onları hatalara karşı toleranslı yapar.

Dereceli bozulma (Graceful degradation) gösterirler. Bir ağ, zaman içerisinde yavaş ve göreceli bir bozulmaya uğrar. Ağlar, problemin ortaya çıktığı anda hemen bozulmazlar.

Dağıtık belleğe sahiptirler. YSA'larında bilgi, ağa dağılmış bir şekilde tutulur. Hücrelerin bağlantı ve ağırlık dereceleri, ağın bilgisini gösterir. Bu nedenle tek bir bağlantının kendi başına anlamı yoktur. Yukarıda çok temel bazı avantajlardan bahsedilmekle beraber, YSA'larının daha pek çok avantajı vardır.

2.12. Yapay Sinir Ağlarının Dezavantajları

YSA'larının, pek çok avantajının yanında bazı dezavantajları da vardır. Belli başlı dezavantajları;

Donanım bağımlıdır. YSA'larının en önemli sorunu donanım bağımlı olmalarıdır. YSA'larının en önemli özellikleri ve varoluş nedenlerinden birisi olan paralel işlem yapabilme yeteneği, paralel çalışan işlemciler ile performans gösterir.

Uygun ağ yapısının belirlenmesinde belli bir kural yoktur. YSA'larında probleme uygun ağ yapısının belirlenmesi için geliştirilmiş bir kural yoktur. Uygun ağ yapısı deneyim ve deneme yanılma yolu ile belirlenmektedir.

Ağın parametre değerlerinin belirlenmesinde belli bir kural yoktur. YSA'larında öğrenme katsayısı, hücre sayısı, katman sayısı gibi parametrelerin belirlenmesinde belirli bir kural yoktur. Bu değerlerin belirlenmesi için belirli bir standart olmamakla birlikte, her problem için farklı bir yaklaşım söz konusu olabilmektedir.

Öğrenilecek problemin ağa gösterimi önemli bir problemdir. YSA'ları nümerik bilgiler ile çalışabilmektedirler. Problemler YSA'larına tanıtılmadan önce nümerik değerlere çevrilmek zorundadırlar. Burada belirlenecek gösterim mekanizması, ağın performansını doğrudan etkileyecektir. Bu da kullanıcının yeteneğine bağlıdır.

Ağın eğitiminin ne zaman bitirilmesi gerektiğine ilişkin belli bir yöntem yoktur. Ağın örnekler üzerindeki hatasının belirli bir değerin altına indirilmesi, eğitimin tamamlandığı anlamına gelmektedir. Burada optimum neticeler veren bir mekanizma henüz yoktur ve YSA'ları ile ilgili araştırmaların önemli bir kolunu oluşturmaktadır.

Ağın davranışları açıklanamamaktadır. Bu sorun YSA'larının en önemli sorunudur. YSA'ları bir probleme çözüm ürettiği zaman, bunun neden ve nasıl olduğuna ilişkin bir ipucu vermez. Bu durum ağa olan güveni azaltıcı bir unsurdur. YSA ve geleneksel algoritmaların karşılaştırılması Çizelge 2.2.'de görüldüğü gibidir.

Çizelge 2.2. Geleneksel Algoritmalar ile YSA'ları

Geleneksel Algoritmalar	Yapay Sinir Ağları
➢ Çıkışlar; koyulan kurallara, girişlerin uygulanması ile elde edilir.	➢ Öğrenme esnasında giriş çıkış bilgileri verilerek, kurallar koyulur.
➢ Hesaplama; merkezi, eş zamanlı ve ardışıldır.	➢ Hesaplama; toplu, eş zamansız ve öğrenmeden sonra paraleldir.
➢ Bellek paketlenmiş ve hazır bilgi depolanmıştır.	➢ Bellek ayrılmış, ve ağa yayılmıştır. Dahilidir.
➢ Hata töleransı yoktur.	➢ Hata töleransı vardır.
➢ Nisbeten hızlıdır.	➢ Yavaş ve donanıma bağımlıdır.
➢ Bilgiler ve algoritmalar kesindir.	➢ Deneyimden yararlanır.

2.13. YSA'larının Kullanıldığı Alanlar

Yapay sinir ağları başlıca; Sınıflandırma, Modelleme ve Tahmin uygulamaları olmak üzere, pek çok alanda kullanılmaktadır. Başarılı uygulamalar incelendiğinde, YSA'larının çok boyutlu, gürültülü, karmaşık, kesin olmayan, eksik, kusurlu, hata olasılığı yüksek sensor verilerinin olması ve problemi çözmek için matematiksel modelin ve algoritmaların bulunmadığı, sadece örneklerin var olduğu durumlarda yaygın olarak kullanıldıkları görülmektedir. Bu amaçla geliştirilmiş ağlar genellikle şu fonksiyonları gerçekleştirmektedirler:

- Muhtemel fonksiyon kestirimleri
- Sınıflandırma
- İlişkilendirme veya örüntü eşleştirme
- Zaman serileri analizleri
- Sinyal filtreleme
- Veri sıkıştırma
- Örüntü tanıma
- Doğrusal olmayan sinyal işleme

- Doğrusal olmayan sistem modelleme
- Optimizasyon
- Kontrol

YSA'ları pek çok sektörde değişik uygulama alanları bulmuştur. Bunlardan bazıları;

Bankacılık: Kredi uygulamaları geliştirilmesi, müşteri analizi ve kredi müracaat değerlendirilmesi, bütçe yatırım tahminleri vs.

Savunma: Silah yönlendirme, hedef seçme, radar, sensor sonar sistemleri, sinyal isleme, görüntü işleme vs.

Elektronik: Kod sırası öngörüsü**,** çip bozulma analizi, non-lineer modelleme vs.

Üretim: Üretim işlem kontrolü, ürün dizaynı, makine yıpranmalarının tespiti, dayanıklılık analizi, kalite kontrolü, iş çizelgeleri hazırlanması vs.

Dil: Sözcük tanıma, yazı ve konuşma çevrimi, dil tercüme vs.

Telekomünikasyon: Görüntü ve data karşılaştırma, filtreleme, eko ve gürültü sönümlendirilmesi, ses ve görüntü işleme, trafik yoğunluğunun kontrolü ve anahtarlama vs.

Güvenlik: Parmak izi tanıma, kredi kartı hileleri saptama, retina tarama, yüz eşleştirme vs.

Görüldüğü gibi YSA'ları günlük hayatımızda farkında olmadığımız pek çok alanda kullanılmaktadır. Gün geçtikçe uygulama alanları genişlemekte ve gelişmektedir (Vural 2008).

3. MATERYAL ve YÖNTEM

Önceki bölümlerde çalışma prensipleri anlatılan yapay sinir ağlarının, bir banka şubesine gelen müşterinin bekleyeceği süreyi tahmin etmek üzere, gün, bekleyen kişi sayısı, bilet türü (bireysel, şirket), kart (0=kartsız, 1=kartlı) değişkenlerini göz önünde bulundurarak nasıl kullanıldığı bu bölümde anlatılmaya çalışılmıştır. Bu amaçla;

1. İlk olarak gerekli veriler bir bankadan alınmıştır.
2. Ağın topolojik yapısı belirlenmiştir. (girdi, ara katman, çıktı katmanı)
3. Ağırlıkların başlangıç değerleri sistem tarafından rastgele atanır daha sonra ağ uygun değerleri öğrenme sırasında kendisi belirler.
4. Ağ eğitimi için verilerden eğitim seti, doğrulama seti, test seti seçilir.

Ağın öğrenmeye başlaması için uygulanan öğrenme kuralına göre ağırlıklar değiştirilerek verilerden (girdi / çıktı değerleri) belirli düzeneğe göre gösterilir ve ağın çıktı değerleri hesaplanır.

Gerçekleşen çıktı değeri ağın ürettiği çıktı değeri karşılaştırılır ve ağın hata değeri hesaplanır.

Geri yayılımlı öğrenme yöntemi ile üretilen hatanın azalması için ağırlıkların değiştirilmesi yapılır.

Çizelge 3.1. Bankadan Alınan Veriler

Gün	Bilet Türü	Kart	Arkada Bekleyen Kişi Sayısı	Bekleme Süresi
Çarşamba	**Bireysel**	**0**	**0**	**0**
Çarşamba	**Bireysel**	**0**	**0**	**0**
Çarşamba	**Bireysel**	**1**	**0**	**0**
Çarşamba	**Bireysel**	**0**	**0**	**4**
Çarşamba	**Bireysel**	**1**	**0**	**1**
Çarşamba	**Bireysel**	**1**	**2**	**1**
Çarşamba	**Bireysel**	**0**	**0**	**1**

Çizelge 3.1. (devam)

Gün	Bilet Türü	Kart	Arkada Bekleyen Kişi Sayısı	Bekleme Süresi
Çarşamba	Bireysel	1	0	3
Çarşamba	Bireysel	1	10	4
Çarşamba	Bireysel	1	10	3
Çarşamba	Bireysel	0	5	9
Çarşamba	Bireysel	0	4	7
Çarşamba	Bireysel	0	3	5
Çarşamba	Bireysel	1	2	0
Çarşamba	Bireysel	1	2	0
Çarşamba	Bireysel	0	3	4
Çarşamba	Bireysel	1	2	1
Çarşamba	Bireysel	0	1	1
Çarşamba	Bireysel	1	3	2
Çarşamba	Bireysel	0	1	2
Çarşamba	Bireysel	1	2	0
Çarşamba	Bireysel	0	2	2
Çarşamba	Bireysel	1	4	1
Çarşamba	Bireysel	1	5	0
Çarşamba	Bireysel	0	0	0
Çarşamba	Bireysel	1	0	0
Salı	Bireysel	1	5	3
Salı	Bireysel	1	4	0
Salı	Bireysel	0	4	4
Salı	Bireysel	0	3	4
Salı	Bireysel	0	0	0
Salı	Bireysel	1	2	0
Salı	Bireysel	0	0	3
Salı	Bireysel	1	0	0
Salı	Bireysel	0	0	0
Salı	Bireysel	0	0	0
Salı	Bireysel	0	0	0
Salı	Bireysel	1	2	0
Salı	Bireysel	1	2	0
Salı	Bireysel	0	0	0
Salı	Bireysel	0	0	0
Salı	Bireysel	0	0	1
Salı	Bireysel	1	1	0
Salı	Bireysel	1	1	0
Salı	Bireysel	1	1	1
Salı	Bireysel	1	0	0
Salı	Bireysel	0	0	0

Çizelge 3.1. (devam)

Gün	Bilet Türü	Kart	Arkada Bekleyen Kişi Sayısı	Bekleme Süresi
Salı	Bireysel	1	3	1
Salı	Bireysel	1	5	1
Salı	Bireysel	1	6	1
Pazartesi	Bireysel	1	1	2
Pazartesi	Bireysel	0	2	3
Pazartesi	Bireysel	1	2	1
Pazartesi	Bireysel	1	9	6
Pazartesi	Bireysel	1	9	0
Pazartesi	Bireysel	1	9	2
Pazartesi	Bireysel	1	10	0
Pazartesi	Bireysel	0	4	9
Pazartesi	Bireysel	1	5	2
Pazartesi	Bireysel	0	2	16
Pazartesi	Bireysel	0	1	5
Pazartesi	Bireysel	0	1	0
Pazartesi	Bireysel	1	3	1
Pazartesi	Bireysel	0	1	6
Pazartesi	Bireysel	0	0	0
Pazartesi	Bireysel	1	5	0
Pazartesi	Bireysel	0	3	6
Pazartesi	Bireysel	1	3	0
Pazartesi	Bireysel	0	2	8
Pazartesi	Bireysel	1	5	2
Cuma	Bireysel	0	11	25
Cuma	Bireysel	1	11	0
Cuma	Bireysel	1	12	5
Cuma	Bireysel	0	13	25
Cuma	Bireysel	1	12	5
Cuma	Bireysel	1	10	3
Cuma	Bireysel	0	9	18
Cuma	Bireysel	0	8	11
Cuma	Bireysel	0	7	11
Cuma	Bireysel	1	4	3
Cuma	Bireysel	0	2	6
Cuma	Bireysel	1	1	1
Cuma	Bireysel	1	10	0
Cuma	Bireysel	0	7	10
Cuma	Bireysel	0	6	10
Cuma	Bireysel	1	6	0
Cuma	Bireysel	0	16	25

Çizelge 3.1. (devam)

Gün	Bilet Türü	Kart	Arkada Bekleyen Kişi Sayısı	Bekleme Süresi
Cuma	Bireysel	1	16	5
Cuma	Bireysel	1	14	2
Cuma	Bireysel	1	11	0
Cuma	Bireysel	1	13	4
Cuma	Bireysel	1	15	3
Cuma	Bireysel	1	13	3
Cuma	Bireysel	1	13	4
Pazartesi	Bireysel	0	5	5
Pazartesi	Bireysel	1	7	1
Pazartesi	Bireysel	1	2	0
Pazartesi	Bireysel	0	9	5
Pazartesi	Bireysel	0	5	6
Pazartesi	Bireysel	1	6	1
Pazartesi	Bireysel	1	4	3
Pazartesi	Bireysel	1	4	4
Pazartesi	Bireysel	0	1	2
Pazartesi	Bireysel	0	2	0
Pazartesi	Bireysel	1	2	0
Pazartesi	Bireysel	1	2	2
Pazartesi	Bireysel	0	0	0
Pazartesi	Bireysel	0	0	0
Pazartesi	Bireysel	0	1	1
Pazartesi	Bireysel	0	2	2
Pazartesi	Bireysel	1	4	0
Pazartesi	Bireysel	1	3	2
Pazartesi	Bireysel	1	8	4
Pazartesi	Bireysel	0	8	4
Pazartesi	Bireysel	0	4	4
Pazartesi	Bireysel	1	4	1
Pazartesi	Bireysel	1	3	0
Pazartesi	Bireysel	0	0	0
Pazartesi	Bireysel	1	1	2
Pazartesi	Bireysel	1	10	0
Cuma	Bireysel	1	7	2
Cuma	Bireysel	1	4	3
Cuma	Bireysel	0	4	7
Cuma	Bireysel	0	4	8
Cuma	Bireysel	0	4	4
Cuma	Bireysel	0	6	5
Cuma	Bireysel	0	5	3

Çizelge 3.1. (devam)

Gün	Bilet Türü	Kart	Arkada Bekleyen Kişi Sayısı	Bekleme Süresi
Cuma	Bireysel	1	4	4
Cuma	Bireysel	0	2	3
Cuma	Bireysel	1	1	0
Cuma	Bireysel	1	3	1
Cuma	Bireysel	0	0	1
Cuma	Bireysel	1	0	0
Cuma	Bireysel	0	0	0
Cuma	Bireysel	1	0	1
Cuma	Bireysel	0	0	1
Cuma	Bireysel	1	2	0
Cuma	Bireysel	1	1	1
Cuma	Bireysel	0	0	0
Cuma	Bireysel	1	3	1
Cuma	Bireysel	0	1	2
Cuma	Bireysel	0	0	2
Cuma	Bireysel	0	0	0
Perşembe	Bireysel	0	3	6
Perşembe	Bireysel	0	2	7
Perşembe	Bireysel	0	1	2
Perşembe	Bireysel	0	4	3
Perşembe	Bireysel	0	0	4
Perşembe	Bireysel	1	0	1
Perşembe	Bireysel	1	7	4
Perşembe	Bireysel	0	7	12
Perşembe	Bireysel	1	6	2
Perşembe	Bireysel	1	5	0
Perşembe	Bireysel	1	5	2
Perşembe	Bireysel	1	5	1
Perşembe	Bireysel	0	5	6
Perşembe	Bireysel	1	0	1
Perşembe	Bireysel	0	0	0
Perşembe	Bireysel	1	2	1
Perşembe	Bireysel	1	3	2
Perşembe	Bireysel	0	8	2
Perşembe	Bireysel	0	12	2
Perşembe	Bireysel	0	9	3
Perşembe	Bireysel	1	1	4
Perşembe	Bireysel	1	1	1
Perşembe	Bireysel	1	5	0
Perşembe	Bireysel	1	0	1

Çizelge 3.1. (devam)

Gün	Bilet Türü	Kart	Arkada Bekleyen Kişi Sayısı	Bekleme Süresi
Perşembe	Bireysel	1	0	1
Perşembe	Bireysel	1	0	0
Cuma	Bireysel	1	1	3
Cuma	Bireysel	0	0	4
Cuma	Bireysel	1	5	7
Cuma	Bireysel	1	3	4
Cuma	Bireysel	0	1	11
Cuma	Bireysel	1	0	0
Cuma	Bireysel	0	4	1
Cuma	Bireysel	0	1	4
Cuma	Bireysel	1	1	0
Cuma	Bireysel	0	0	0
Cuma	Bireysel	0	1	2
Cuma	Bireysel	1	9	3
Cuma	Bireysel	0	10	16
Cuma	Bireysel	0	4	22
Cuma	Bireysel	0	6	24
Cuma	Bireysel	1	6	2
Cuma	Bireysel	1	12	7
Cuma	Bireysel	1	11	7
Cuma	Bireysel	0	9	8
Cuma	Bireysel	0	6	4
Pazartesi	Şirket	1	28	43
Pazartesi	Şirket	1	27	43
Pazartesi	Şirket	1	28	40
Pazartesi	Şirket	0	27	41
Pazartesi	Şirket	1	26	39
Pazartesi	Şirket	1	4	30
Salı	Şirket	1	20	1
Salı	Şirket	1	22	0
Salı	Şirket	1	24	3
Salı	Şirket	1	26	5
Salı	Şirket	1	25	4
Salı	Şirket	1	20	22
Salı	Şirket	1	23	1
Salı	Şirket	0	22	8
Salı	Şirket	0	16	26
Salı	Şirket	1	13	6
Salı	Şirket	1	9	4
Salı	Şirket	0	9	24

Çizelge 3.1. (devam)

Gün	Bilet Türü	Kart	Arkada Bekleyen Kişi Sayısı	Bekleme Süresi
Salı	**Şirket**	**0**	**8**	**24**
Salı	**Şirket**	**1**	**8**	**0**
Salı	**Şirket**	**0**	**9**	**17**
Salı	**Şirket**	**1**	**10**	**0**
Salı	**Şirket**	**1**	**11**	**0**
Salı	**Şirket**	**1**	**10**	**0**
Salı	**Şirket**	**1**	**0**	**0**
Salı	**Şirket**	**0**	**0**	**0**
Salı	**Şirket**	**1**	**0**	**0**
Salı	**Şirket**	**1**	**0**	**0**
Salı	**Şirket**	**1**	**1**	**0**
Salı	**Şirket**	**1**	**0**	**2**
Salı	**Şirket**	**1**	**0**	**0**
Salı	**Şirket**	**1**	**1**	**0**
Salı	**Şirket**	**1**	**0**	**0**
Salı	**Şirket**	**1**	**0**	**0**
Salı	**Şirket**	**1**	**1**	**4**
Salı	**Şirket**	**1**	**0**	**5**
Salı	**Şirket**	**1**	**0**	**0**
Salı	**Şirket**	**1**	**0**	**0**
Salı	**Şirket**	**1**	**0**	**0**
Çarşamba	**Şirket**	**1**	**0**	**0**
Çarşamba	**Şirket**	**1**	**0**	**0**
Çarşamba	**Şirket**	**1**	**0**	**0**
Çarşamba	**Şirket**	**1**	**0**	**0**
Çarşamba	**Şirket**	**1**	**0**	**0**
Çarşamba	**Şirket**	**1**	**0**	**0**
Çarşamba	**Şirket**	**1**	**1**	**0**
Çarşamba	**Şirket**	**1**	**0**	**1**
Çarşamba	**Şirket**	**1**	**0**	**0**
Çarşamba	**Şirket**	**1**	**1**	**0**
Çarşamba	**Şirket**	**1**	**0**	**1**
Çarşamba	**Şirket**	**1**	**0**	**0**
Çarşamba	**Şirket**	**1**	**0**	**0**
Çarşamba	**Şirket**	**1**	**0**	**0**
Çarşamba	**Şirket**	**1**	**0**	**0**
Çarşamba	**Şirket**	**1**	**0**	**0**
Çarşamba	**Şirket**	**1**	**0**	**0**
Çarşamba	**Şirket**	**1**	**0**	**0**
Çarşamba	**Şirket**	**0**	**0**	**0**

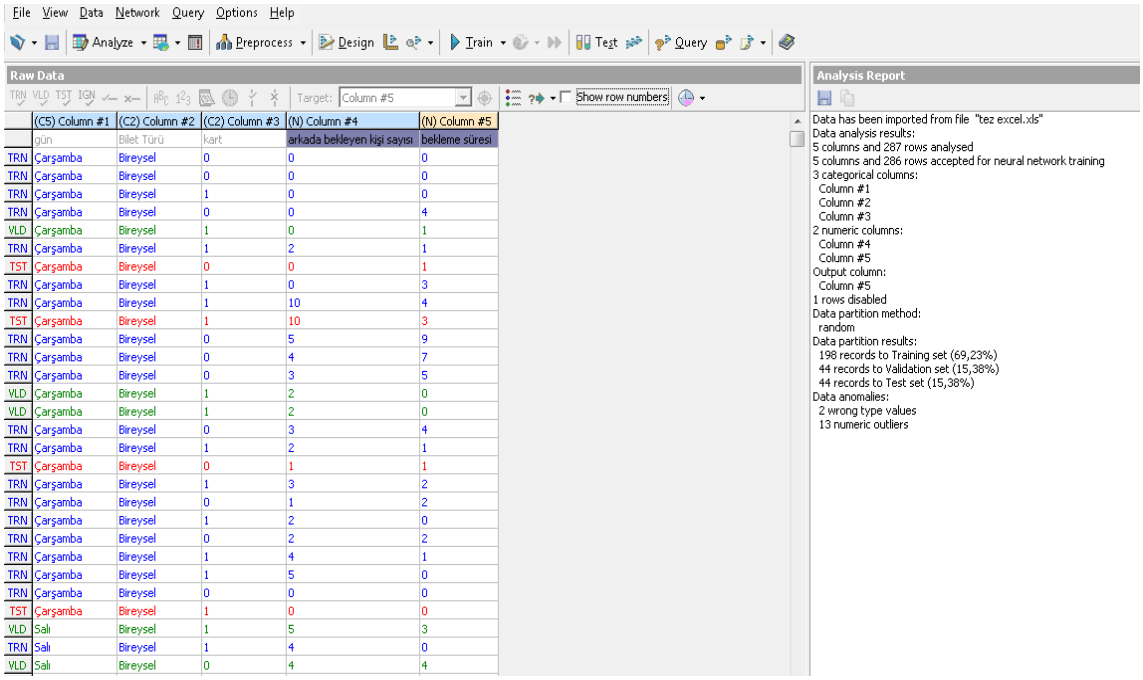
Çizelge 3.1. (devam)

Gün	Bilet Türü	Kart	Arkada Bekleyen Kişi Sayısı	Bekleme Süresi
Çarşamba	Şirket	1	1	1
Çarşamba	Şirket	0	0	3
Çarşamba	Şirket	1	0	0
Cuma	Şirket	1	0	0
Cuma	Şirket	1	0	0
Cuma	Şirket	1	0	0
Cuma	Şirket	1	0	0
Cuma	Şirket	1	0	0
Cuma	Şirket	1	0	0
Cuma	Şirket	1	0	0
Cuma	Şirket	1	1	2
Cuma	Şirket	1	0	3
Cuma	Şirket	1	0	3
Cuma	Şirket	1	0	0
Cuma	Şirket	1	0	2
Cuma	Şirket	0	1	0
Cuma	Şirket	1	4	5
Cuma	Şirket	1	3	6
Cuma	Şirket	1	2	1
Cuma	Şirket	1	1	2
Cuma	Şirket	1	2	0
Cuma	Şirket	0	2	23
Cuma	Şirket	1	1	2
Cuma	Şirket	1	0	0
Perşembe	Şirket	1	1	0
Perşembe	Şirket	1	0	1
Perşembe	Şirket	1	0	0
Perşembe	Şirket	1	0	0
Perşembe	Şirket	1	0	0
Perşembe	Şirket	1	0	0
Perşembe	Şirket	1	0	0
Perşembe	Şirket	1	0	4
Perşembe	Şirket	1	1	0
Perşembe	Şirket	1	0	2
Perşembe	Şirket	1	0	0
Perşembe	Şirket	1	1	0
Perşembe	Şirket	0	0	1
Perşembe	Şirket	1	0	0
Perşembe	Şirket	1	1	0

Çizelge 3.1.'de görüldüğü gibi uygulamada kullanılmak üzere 286 tane veri toplanmıştır. Kullanılan program Alyuda NeuroIntelligence, ağın öğrenme algoritması ise danışmanlı öğrenmedir. Ağın eğitimi için girdi katmanında 4 girdi elemanı vardır. Bunlar;

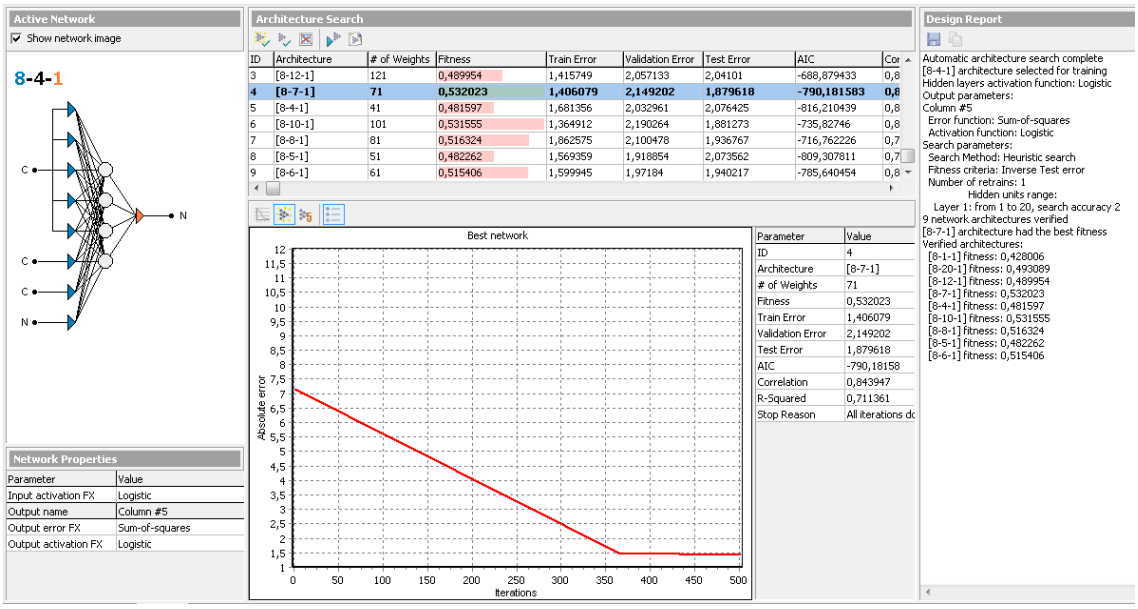
1. Gün (Pazartesi, Salı, Çarşamba, Perşembe, Cuma)
2. Bilet Türü (Bireysel, Şirket)
3. Kart (0, 1)
4. Arkada Bekleyen Kişi Sayısı'dır.

Ayın belirli günlerinden veriler alınmıştır. Bilet türü bireysel ve şirket olarak ayrılmıştır çünkü verilerin alındığı banka sisteminde bireysel müşteriler ve şirket müşteriler kendi aralarında ayrı ayrı değerlendirilmekte ve her ikisi için ayrı bir numara üretmektedir. Ayrıca müşteriler kartlı ve kartsız olarak da ayrılmaktadır. Sisteme dâhil olan müşteri eğer kartı ile sıra almışsa sistem otomatik olarak bu müşteriye öncelik tanımaktadır. Arkada bekleyen kişi sayısı ise; müşteri hizmet aldığı sırada arkada bekleyen kişi sayısını göstermektedir. Çıktı katmanında ise bekleme süresi vardır. Gizli katman sayısını sistem deneyerek en uygun olan dizaynı bulmaktadır. İlk olarak Alyuda NeuroIntelligence programına verilerimizi aktardık. Veri dizisi Şekil 3.1'de verilmiştir.



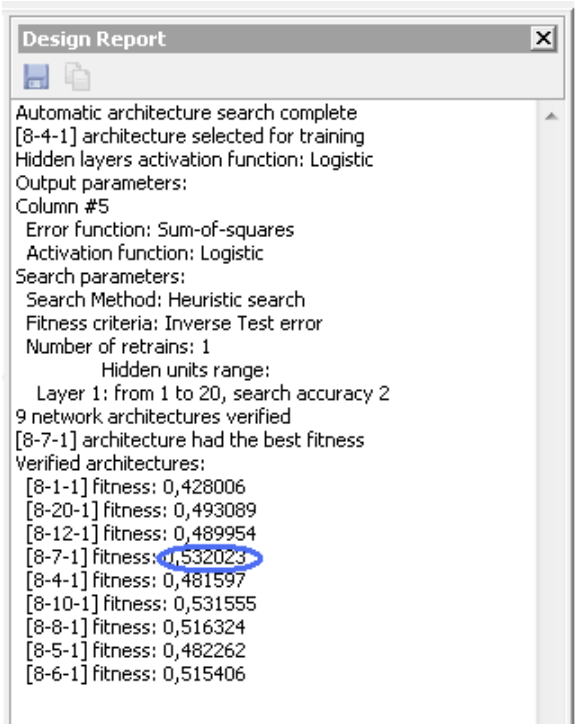
Şekil 3.1. Veri Dizisi

Sistem tarafından 198 kayıt eğitim seti, 44 kayıt doğrulama seti, 44 kayıt ise test seti olarak seçilmiştir.



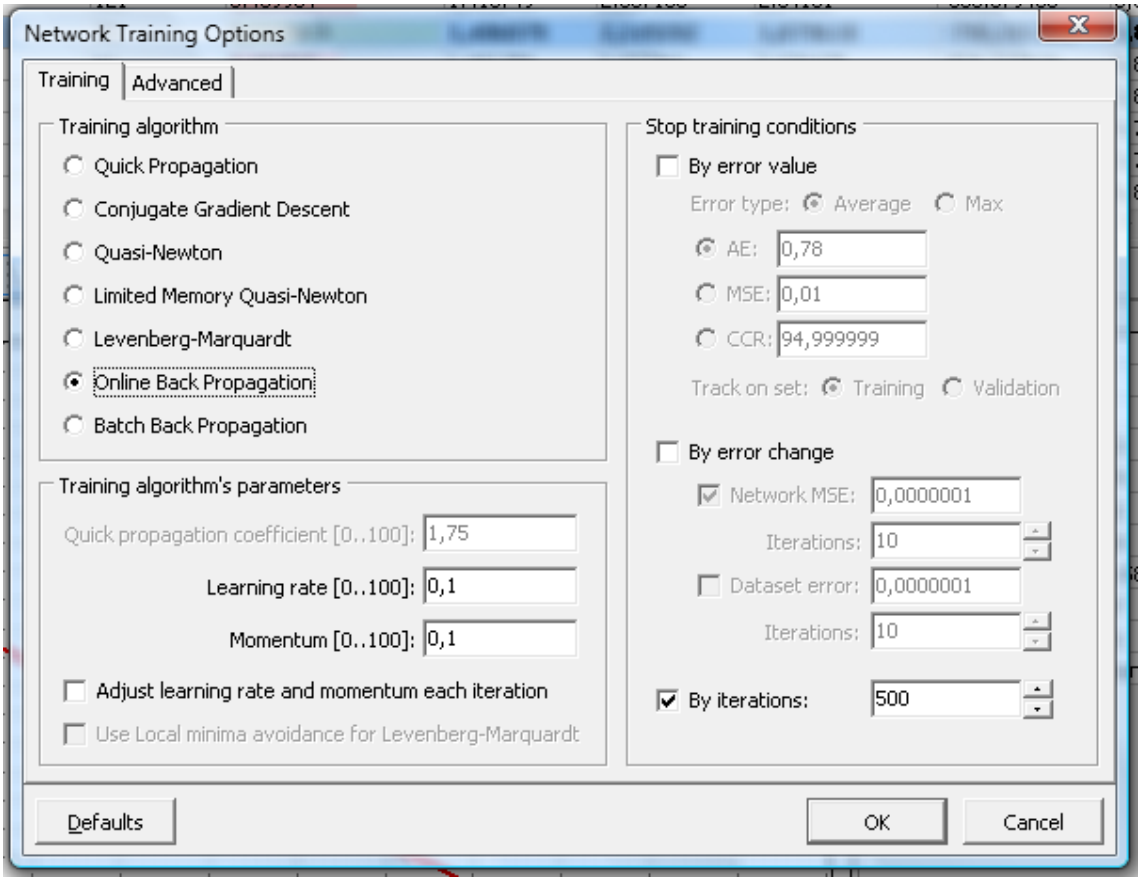
Şekil 3.2. Sistem Mimarisi

Şekil 3.3’ de görüldüğü gibi sistem için en uygun yapı 8-7-1’dir.



Şekil 3.3. Dizayn Raporu

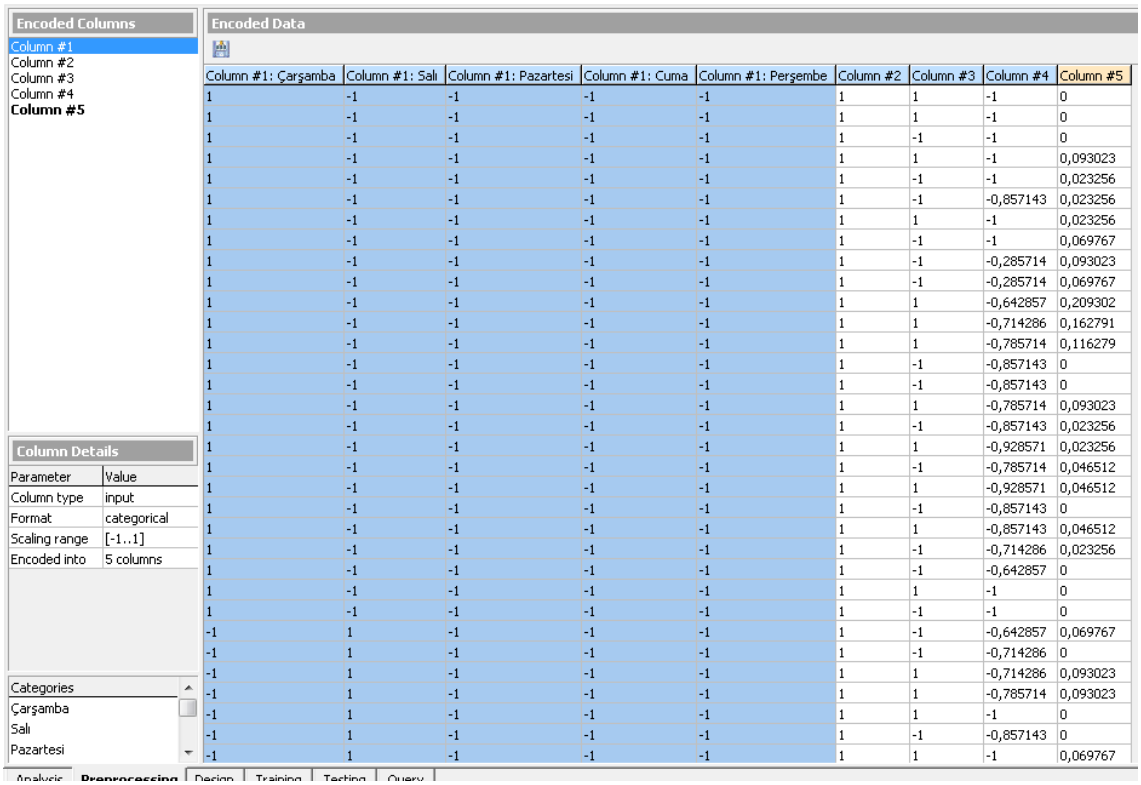
Sistemde gerekli değişiklikler yapıldıktan sonra eğitime geçilmiştir. Sistem eğitim seçenekleri Şekil 3.4’deki gibidir.



Şekil 3.4. Sistem Eğitim Seçenekleri

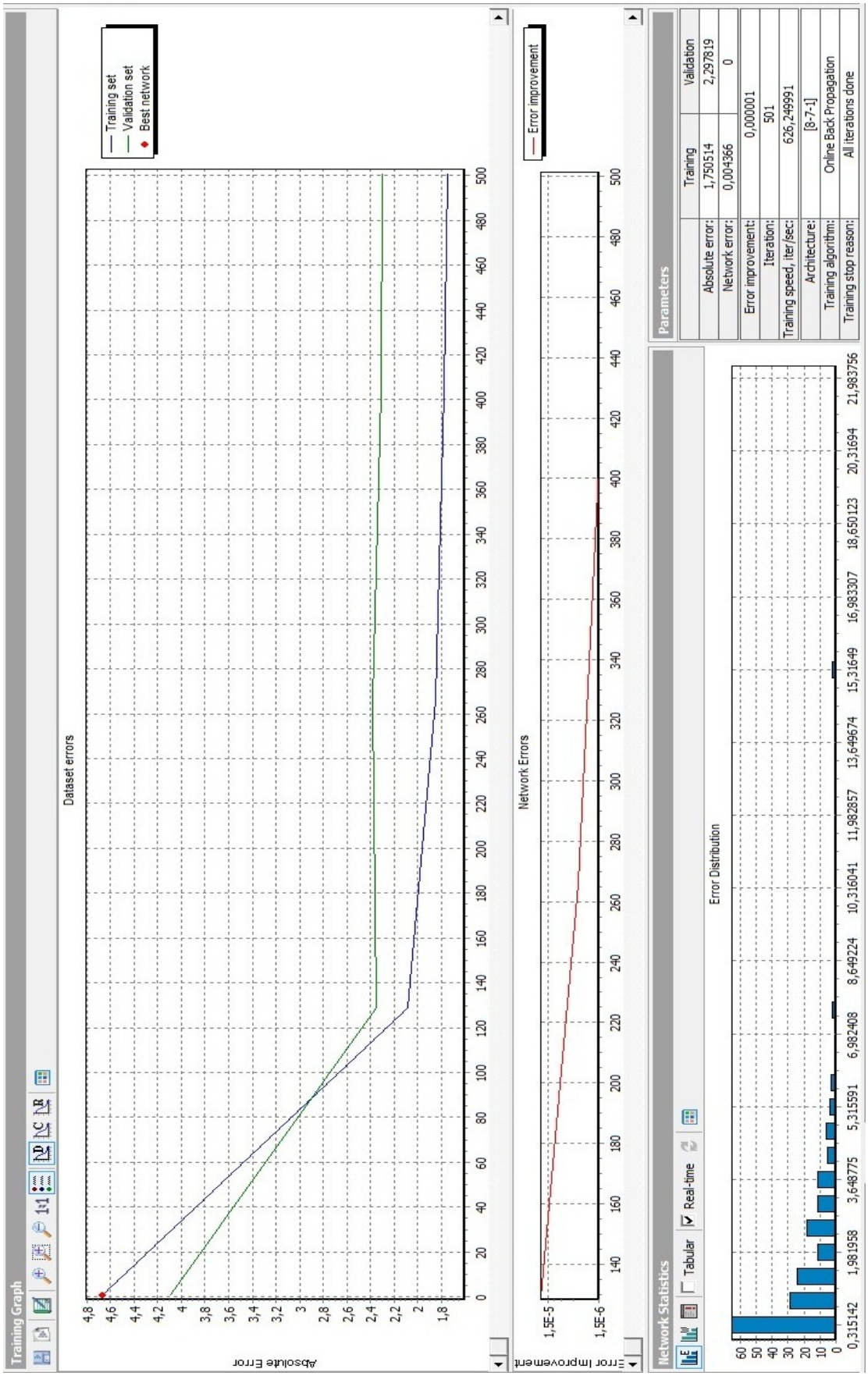
4. ARAŞTIRMA BULGULARI ve TARTIŞMA

Eğitim algoritması geri yayılımlı ve 500 iterasyona kadar sistemi eğitmesi yönünde seçilmiştir. Öğrenme katsayısı, momentum katsayısı ve öğrenme algoritması gibi sistem parametrelerini değiştirilerek en iyi sonucu bulmak için denemeler yapılmıştır. Şekil 4.1'de girmiş olduğumuz veriler sistem tarafından [-1…1] aralığında değerler verilmiştir. Bu değerleri kullanması sistemin aktivasyon fonksiyonu olarak hiperbolik tanjant fonksiyonunu kullandığını göstermektedir.



Şekil 4.1. Sütunların Kodlanmış Halleri

Denediğimiz parametre değerlerinin (momentum katsayısı, öğrenme katsayısı, öğrenme algoritması) vermiş olduğu hata değerleri aşağıdaki gibidir:



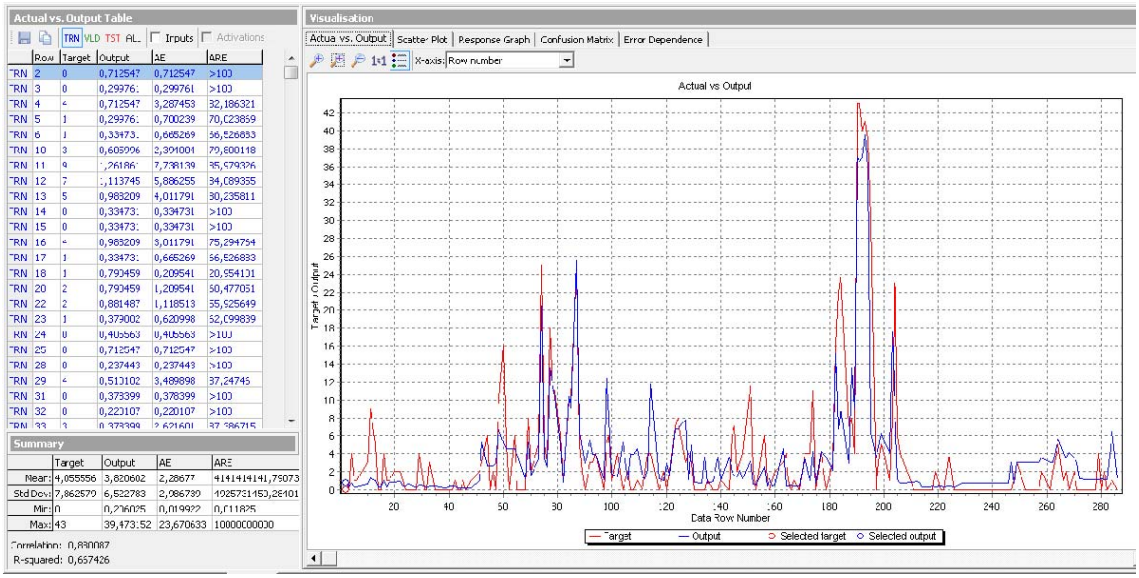
Şekil 4.2. Eğitim Grafiği

Parameters		
	Training	Validation
Absolute error:	1,750514	2,297819
Network error:	0,004366	0
Error improvement:	0,000001	
Iteration:	501	
Training speed, iter/sec:	626,249991	
Architecture:	[8-7-1]	
Training algorithm:	Online Back Propagation	
Training stop reason:	All iterations done	

Şekil 4.3. Eğitim Grafiği Parametreleri

Görüldüğü gibi eğitimde mutlak hata 1,75051, doğrulama setinde ise mutlak hata 2,2978 olarak bulunmuştur.

Hata gelişimi 0.000001, 501 iterasyon, yapısı 8-7-1, eğitim algoritması geri yayılımlı ve son veriye kadar hepsi uygulanmış eğitim durdurulmuştur.



Şekil 4.4. Gerçek Çıktı Değerleri ile Sistem Çıktı Değerleri

Gün	Bilet Türü	Kart	Arkada Bekleyen Kişi Sayısı	Bekleme Süresi
Pazartesi	Bireysel	0	10	14,73645
Pazartesi	Bireysel	1	10	2,698126
Pazartesi	Şirket	1	10	13,19148
Pazartesi	Şirket	0	10	32,94325
Salı	Bireysel	0	10	0,993812
Salı	Bireysel	1	10	0,542115
Salı	Şirket	1	10	1,090035
Salı	Şirket	0	10	3,721586
Çarşamba	Bireysel	0	10	2,294521
Çarşamba	Bireysel	1	10	0,839092
Çarşamba	Şirket	1	10	2,27552
Çarşamba	Şirket	0	10	11,72044
Perşembe	Bireysel	0	10	4,024446
Perşembe	Bireysel	1	10	1,132181
Perşembe	Şirket	1	10	3,833963
Perşembe	Şirket	0	10	18,84205
Cuma	Bireysel	0	10	15,26818
Cuma	Bireysel	1	10	2,477059
Cuma	Şirket	1	10	13,45361
Cuma	Şirket	0	10	32,87384

Şekil 4.5. Sonuç Tablosu

Şekil 4.5'de görüldüğü gibi girdiler manuel olarak sisteme girilmiştir ve yoğun olan günlerde bekleme süresinin fazla olacağı çıktılarda görülmüştür.

Online Back Propagation, Conjugate Gradient Descent, Quasi-Newton, Levenberg-Marquardt olmak üzere dört tane öğrenme algoritması öğrenme katsayısı ve momentum katsayısı değiştirilerek denenmiştir. Öğrenme katsayısını 0.2-0.4 arasında ve momentum katsayısını da 0.6-0.8 arasında alınıp sonuçlar karşılaştırılmıştır.

Öğrenme algoritmalarına göre sistem çıktıları Şekil 4.6'da görüldüğü gibidir.

Öğrenme Algoritması:	Öğrenme Katsayısı:	Momentum Katsayısı:	Öğrenmede Setinde Toplam Hata:	Doğrulamada Setinde Toplam Hata:
Conjugate Gradient Descent	0,2	0,1	1,3074	2,3855
Conjugate Gradient Descent	0,3	0,1	1,404	2,096
Conjugate Gradient Descent	0,4	0,1	1,3823	2,8287
Conjugate Gradient Descent	0,2	0,6	1,3315	3,267
Conjugate Gradient Descent	0,2	0,7	1,4151	2,5967
Conjugate Gradient Descent	0,2	0,8	1,3631	2,8898
Conjugate Gradient Descent	0,3	0,6	1,3412	2,7548
Conjugate Gradient Descent	0,3	0,7	1,3655	2,3853
Conjugate Gradient Descent	0,3	0,8	1,4049	2,7634
Conjugate Gradient Descent	0,4	0,6	1,3306	2,6284
Conjugate Gradient Descent	0,4	0,7	1,4039	2,3359
Conjugate Gradient Descent	0,4	0,8	1,4065	2,2048
Quasi-Newton	0,2	0,1	1,4236	3,3022
Quasi-Newton	0,3	0,1	1,3605	3,6691
Quasi-Newton	0,4	0,1	1,4048	2,4862
Quasi-Newton	0,2	0,6	1,5461	2,2391
Quasi-Newton	0,2	0,7	1,422	3,6434
Quasi-Newton	0,2	0,8	1,4467	2,4469
Quasi-Newton	0,3	0,6	1,488	2,7704
Quasi-Newton	0,3	0,7	1,3898	3,325
Quasi-Newton	0,3	0,8	1,6533	2,0015
Quasi-Newton	0,4	0,6	1,4662	2,9428
Quasi-Newton	0,4	0,7	1,3973	2,2093
Quasi-Newton	0,4	0,8	1,5903	2,7406
Levenberg-Marquardt	0,2	0,1	12,5296	12,5314
Levenberg-Marquardt	0,3	0,1	16,756	16,7112
Levenberg-Marquardt	0,4	0,1	2,5401	3,3075
Levenberg-Marquardt	0,2	0,6	3,6408	3,7344
Levenberg-Marquardt	0,2	0,7	2,0553	2,7986
Levenberg-Marquardt	0,2	0,8	1,9694	3,069
Levenberg-Marquardt	0,3	0,6	3,1705	3,3632
Levenberg-Marquardt	0,3	0,7	23,3508	23,5479
Levenberg-Marquardt	0,3	0,8	2,0457	2,7521
Levenberg-Marquardt	0,4	0,6	1,9906	2,4644
Levenberg-Marquardt	0,4	0,7	2,1638	3,0475
Levenberg-Marquardt	0,4	0,8	15,3598	15,3551
Online Back Propagation	0,2	0,1	1,9639	2,1894
Online Back Propagation	0,3	0,1	1,8627	2,0823
Online Back Propagation	0,4	0,1	1,846	2,0724
Online Back Propagation	0,2	0,6	1,8685	2,1378
Online Back Propagation	0,2	0,7	1,7743	2,6696
Online Back Propagation	0,2	0,8	1,7702	2,629
Online Back Propagation	0,3	0,6	1,7454	2,4027
Online Back Propagation	0,3	0,7	1,7286	2,5536
Online Back Propagation	0,3	0,8	1,658	2,5728
Online Back Propagation	0,4	0,6	1,7529	2,6269
Online Back Propagation	0,4	0,7	1,7846	2,6725
Online Back Propagation	0,4	0,8	1,7448	2,4108

Şekil 4.6. Öğrenme Algoritmalarına Göre Sistem Çıktıları

5. SONUÇ

Günlük hayatta gerçekleştirilen birçok eylemde, kuyrukta bekleme sorunu ile karşılaşılmaktadır. Banka işlemi, call center gibi müşterinin şirketlerle ilişkilerinde hep bir bekleme zorunluluğu ortaya çıkmaktadır. Hizmet almak için herhangi bir servis sistemini tercih eden müşterinin, ortalama bekleme süresini bilmeden beklemeleri şikâyetlere sebep olmaktadır.

Örneğin müşteri işlem zamanını beklemek yerine çok fazla yığılma olduğu günlerde hizmet alanını terk ederek diğer işlemleriyle ilgilenmeyi tercih etmektedir. Daha sonrasında hizmet alanına geri döndüğünde sırası geçmiş olmaktadır veya yeniden beklemek durumunda kalmaktadır. Her iki durumda müşteri şikâyetine sebebiyet vermektedir.

Banka gibi servis sistemlerinde müşterilerin işlemlerini tamamlamadan sistemden ayrılmaları genel olarak kuyruktaki yığılmalara bağlıdır.

Uygulama yeri olarak seçilen banka şubesinde gelen müşterinin ortalama ne kadar bekleyeceğini tahmin eden bir sistem kurularak; müşteri bekleyeceği süreden haberdar olarak bekleyecek veya başka işi varsa o süre içerisinde başka işleri ile ilgilenmesi mümkün olacaktır.

Uygulamada Alyuda NeuroIntelligence programı kullanılmıştır. Bankadan 286 tane veri toplanmıştır. Verilerin alındığı bankada müşteriler bireysel ve şirket müşterileri olarak ikiye ayrılmakta ayrıca bireysel ve şirket müşterileri de kendi aralarında kartlı ve kartsız müşteriler olarak ayrılmaktadır. Sıra almak için gelen müşteriye eğer kartlı müşteri ise sistem tarafından öncelik tanınmaktadır. Kartsız müşteri kartlı müşteriler bitene kadar sistemde beklemektedir. Ayrıca bankada günlere göre de yoğunluk yaşanmaktadır. Örneğin alınan verilere bakıldığında genel itibariyle pazartesi ve cuma günleri yoğunluk yaşanmaktadır ve alınan verilerde şirket müşterilerinin bireysel müşterilere göre işlem

sürelerinin daha fazla olduğu görülmektedir. Uygulamada pazartesi, salı, çarşamba, perşembe, cuma, kartlı, kartsız, arkada bekleyen kişi sayısı girdi elemanı olarak belirlenmiştir. Ağın mimari yapısı program tarafından denenerek 8 girdi elemanı 7 ara katman 1 tanede çıktı elemanı olarak belirlenmiştir. Uygulamada kullandığımız ağ yapısı çok katmanlı algılayıcıdır. Öğrenme eğitme algoritması olarak genelde geri yayılım algoritması tercih edilir. Ağda 286 tane verinin 198 tanesi eğitim seti, 44 tanesi doğrulama seti ve 44 tanesi de test seti olarak otomatik olarak ayrılmıştır. Uygulamada kullandığımız tüm veriler program tarafından (1..-1) arasında numaralandırılmıştır. 500 iterasyona kadar sistem eğitilmiştir. Eğitim sonucunda programın öğrenip öğrenmediğini test etmek için manüel olarak birkaç deneme yapılmıştır. Örneğin pazartesi günü bireysel ve kartsız sıra alan bir müşteri ve önünde bekleyen 10 kişi varsa 14,7 dakika bekleyecektir veya yine bireysel ve kartlı sıra alan bir müşteri önünde bekleyen kişi sayısı 10 ise bekleme süresi 2,6 dakika olacaktır. Şirket müşterisi için de bir deneme yapacak olursak, pazartesi günü şirket ve kartlı sıra alan bir müşteri önünde bekleyen kişi sayısı 10 ise 13,19 dakika beklerken kartsız sıra alan şirket müşterisi 32 dakika bekleyecektir. Görüldüğü gibi kartsız sıra alan müşteriler kartlı müşterilere göre daha fazla süre beklemektedirler. Aynı şekilde şirket müşterilerinin bireysel müşterilere göre daha fazla işlemleri olduğundan da fazla süre bekledikleri görülmektedir. Bu sonuçlar alınan veriler ile karşılaştırıldığında anlamlı sonuç ürettiği görülmektedir.

Bu sistem ile elimizdeki verilerle kuyrukta bekleme süresi tahmin edilmeye çalışılmıştır. Sistem, kuyrukta bekleme sorunu olan tüm kurumlarda bekleme süresi tahmininde kullanılabilir.

KAYNAKLAR

Aksungur S., 2009, Üç Serbestlik Dereceli Bir Robotun Yapay Sinir Ağları ve Genetik Algoritma Kullanılarak Engelli Ortamda Çarpışmasız Yörünge Planlaması, Yüksek Lisans Tezi, Selçuk Üniversitesi, Konya.

Atik, K., 2006. Yapay Zeka Uygulamalarında "Yapay Sinir Ağları"nın Tesisat Mühendisliğinde Kullanımı. Tesisat Mühendisliği Dergisi, 8 (91), 83-87.

Baş N., 2006, Yapay Sinir Ağları Yaklaşımı ve Bir Uygulama, Yüksek Lisans Tezi, Mimar Sinan Güzel Sanatlar Üniversitesi, İstanbul.

Bayındır, R. ve Sesveren, Ö., 2008. YSA Tabanlı Sistemler İçin Görsel Bir Arayüz Tasarımı. Pamukkale Üniversitesi Mühendislik Fakültesi Mühendislik Bilimleri Dergisi, 14 (1), 101-109.

Burmaoğlu, S. , 2009. Birleşmiş Milletler Kalkınma Programı Beşeri Kalkınma Endeksi Verilerini Kullanarak Diskriminant Analizi, Lojistik Regresyon Analizi ve Yapay Sinir Ağlarının Sınıflandırma Başarılarının Değerlendirilmesi. Doktora Tezi, Sosyal Bilimler Enstitüsü, Erzurum.

Ekinci, E., Temur, G.T., Çelebi, D. ve Bayraktar, D., 2008. Ekonomik Kriz Döneminde Firma Başarısı Tahmini: Yapay Sinir Ağları Tabanlı Bir Yaklaşım, Endüstri Mühendisliği Dergisi, 21 (1), 17-29.

Erdem, O.A. ve Uzun, E., 2005. Yapay Sinir Ağları ile Türkçe Times New Roman Arial ve El Yazısı Karakterlerini Tanıma. Gazi Üniversitesi Mühendislik Mimarlık Fakültesi Dergisi, 20 (1), 13-19.

Ergezer, H., Dikmen, M. ve Özdemir, E., 2003. Yapay Sinir Ağları ve Tanıma Sistemleri. Bilgisayar Mühendisliği, Başkent Üniversitesi, http://www.elyadal.org/pivolka/06/yapay.htm (03.11.2010).

Ersöz, E., Yaman, N. ve Birgören, B., 2008. Müşteri İlişkileri Yönetiminde Verilerin Yapay Sinir Ağları İle Modellenmesi ve Analizi. Gazi Üniversitesi Mühendislik Mimarlık Fakültesi Dergisi, 23 (4), 759-767.

Güler, İ. ve Übeyli, E.D., 2006. Çok Katmanlı Perseptron Sinir Ağları ile Diyabet Hastalığının Teşhisi. Gazi Üniversitesi Mühendislik Mimarlık Fakültesi Dergisi, 21 (2), 319-326.

Hamzaçelebi, C. ve Kutay, F., 2004. Yapay Sinir Ağları ile Türkiye Elektrik Enerjisi Tüketiminin 2010 Yılına Kadar Tahmini. Gazi Üniversitesi Mühendislik Mimarlık Fakültesi Dergisi, 19 (3), 227-233.

Kaçar Durmuş, H. ve Meriç, C., 2005. Makine Mühendisliğinde Yapay Sinir Ağlarının (YSA) Kullanımı. Mühendis ve Makina, 46(546), 47-56.

Kaya, İ. ve Engin, O., 2005. Kalite İyileştirme Sürecinde Yapay Zeka Tekniklerinin Kullanımı. Pamukkale Üniversitesi Mühendislik Fakültesi Mühendislik Bilimleri Dergisi, 11 (1), 103-114.

Öztemel, E. , 2003. Yapay Sinir Ağları. Papatya Yayıncılık, 232s, İstanbul.

Vural Y., 2008, Kredi Kartı İçin Yapay Sinir Ağları Uygulaması, Yüksek Lisans Tezi, Haliç Üniversitesi, İstanbul

Yıldız Ö., 2006, Döviz Kuru Tahmininde Yapay Sinir Ağlarının Tahmini, Yüksek Lisans Tezi, Eskişehir Osman Gazi Üniversitesi, Eskişehir

Yurtoğlu H. , 2005, Yapay Sinir Ağları Metodolojisi ile Öngörü Modellemesi: Bazı Makroekonomik Değişkenler İçin Türkiye Örneği, Uzmanlık Tezleri, DPT, Ankara

ÖZGEÇMİŞ

Doğum Tarihi: 27.06.1984

Doğum Yeri: Merkez / Erzurum

Öğrenim Bilgileri:

Lisans: 2001-2005 Sakarya Üniversitesi Endüstri Mühendisliği Bölümü

Yüksek Lisans: 2006-2010 Atatürk Üniversitesi Endüstri Mühendisliği Bölümü

Çalıştığı Kurumlar:

2007-2008 Akbank T.A.Ş. Lalapaşa Şubesi Operasyon Bölümünde Yetkili 4 olarak görev yaptı.

2009-2010- T.C. Ziraat Bankası Erzurum Şubesi Krediler Servisinde Servis Görevlisi olarak görev yaptı.

2010 Kars İl Afet ve Acil Durum Müdürlüğü'nde Endüstri Mühendisi olarak görev yapmakta.