**Özet**

Xox oyunu bütün yaştan insanların kendi aralarında eğlenmek için oynadıkları basit kuralları olan, iki kişi ile oynanan ve çoğu yerde ismi “Tic Tac Toe” olarak geçen üçleme oyunlarındandır. UCI Machine Learning Repository’ den alınan verilere göre oluşturulan yapay sinir ağımızda (oyuna başlayan karakter olarak x kabul edilmiştir) x karakterinin girilen özniteliklere göre kazanma durumunu incelemektedir. Alınan örneklerde özniteliklerin alabileceği değerler “x”, “o” veya “b” (boş) olmak üzere 3 adettir. 3x3 lük oyun alanında girilen öznitelikler göz önünde bulundurularak, 3x3 lük oyun alanında yatay veya dikey olarak herhangi bir yerde yan yana x gelmesi durumunda x karakteri kazanmış kabul edilmekte ve alınan örneğin sınıfı positive (x kazanmış) olarak kabul edilmektedir. Kullanılan veri seti göz önünde bulundurularak oluşturulan yapay sinir ağı bir eğitimden geçirilmektedir. Ardından sonucu belli olmayan örneklerin sınıflarını belirlemek amaçlanmıştır. Tahmin işi, tasarlanan yapay sinir ağına yaptırılmış ve doğruluk oranı %82 olarak bulunmuştur.

**1.Giriş**

Günümüzdeki sinir ağı teorisinin temelini nörolog olan Donald Hebb oluşturduğu bilinmektedir. İnsan vücudunun en önemli birimi olan beynin nasıl çalıştığını ve nasıl öğrendiğini anlamak için çalışmalarda bulunmuştur. Hebb çalışmalarına beynin en temel yapısını oluşturan sinir hücresini inceleyerek başlamıştır. İki sinir hücresinin birbirleriyle nasıl bir ilgileşim sergilediklerini incelemiş ve sinir ağı teorisini bu temel üzerine oturtmuştur. Bu temel kuşkusuz tek gerçek değildir. Çünkü beynin nasıl bir çalışma sergilediği şu an dahi teoriler yardımıyla açıklanmaktadır. Ancak Hebb ’in yardımıyla bu fikir ile yola çıkılmış ve günümüzdeki yüzlerce ayrı teoriyle geniş bir yelpazeye hitap eder hale gelmiştir. Şu an gerçek yaşamda kullanılan ve başarı oranı %99’lar ile ifade edilen bir sürü yapay sinir ağı (YSA) modeli mevcuttur. Tüm geliştirilen modeller bilgisayar dünyasında “çözümsüz” veya “karmaşık” olarak nitelendirilen problemlerin çözümünü hedeflemekte ve hatta bir kısmını başarıyla çözmektedir [1,2].

1993 yılında David F. Bogel tarafından gerçekleştirilmiş olan bu çalışmada Tic-Tac-Toe oyununda verilen girişlere göre x in kazanma durumunu saptayabilmek için bir yapay sinir ağı modellenmiştir. Oluşturulan bu yapay sinir ağında girişler ve çıkışlar için 9 adet nöron kullanılmıştır. Oluşturulan senaryoya göre ilk başlayan oyuncu x karakterini oynamakta ve ikinci oyuncu ise o karakterinin oynamaktadır. İlk olarak 1 adet gizli katman seçilmiştir ve öğrenme katsayısı ve momentum katsayısı 0,5 olarak seçilmiştir. Ara katman sayısı 50 olacak şekilde seçilmiştir. Aktivasyon fonksiyonu ise sigmoid olarak belirlenmiştir. Bu durumda sonuç olarak %93 civarı bir başarıya ulaşılmıştır[3].

2005 yılında Mehmet Çuhadar, Cihan Kayacan tarafından gerçekleştirilen bu çalışmada, yapay sinir ağı kullanılarak Türkiye’deki bakanlık belgeli konaklama işletmelerindeki, dış turizm talebi ile oluşan doluluk oranları tahmin edilmiştir. Çalışmada Turizm Bakanlığı tarafından yıllık olarak yayınlanan “Konaklama İstatistikleri” bültenlerinden elde edilen “1990 – 2002” yıllarına ait aylık verilerden yararlanılmıştır. Çalışma neticesinde gerçek değerlere çok yakın tahmin değerleri ve çok düşük hata oranları elde edilmiştir. Çalışmanın Türk turizm literatürüne katkısı, konaklama işletmelerinde doluluk oranlarının tahmininde yeni bir yöntemi bir uygulama ile göstermektir. İleriye yönelik yapılacak çalışmalarda, yapay sinir ağları ile farklı değişkenleri kullanarak belirli bir bölge veya il sınırları içerisinde faaliyet gösteren konaklama işletmelerindeki doluluk oranları, yerli ve yabancı konukların konaklama işletmelerindeki harcamaları, turistlerin geceleme sayıları ve ortalama kalış süreleri tahmin edilebilir[4].

2005 yılında Cemil Çolak, M. Cengiz Çolak, M. Ali Atıcı gerçekleştirilen bu çalışmada, ateroskleroz’un tahmin edilebilmesi için bir yapay sinir ağı oluşturulması amaçlanmıştır.

**Gereç ve Yöntem:** Haziran 2003 ile Kasım 2003 tarihleri arasında, kesikli ve sürekli değişkenlerden oluşan yirmi adet klinik parametre, radial arterde ateroskleroz saptanan on hasta ile radial arterde ateroskleroz saptanmayan on beş hastadan elde edilmiştir. Yapay sinir ağları, ateroskleroz verilerine uygulanmıştır.

**Bulgular:** Geliştirilen yapay sinir ağının toplam ayrımsama oranı, test verisinde % 80 olarak bulunmuştur.

**Sonuç:** Yapay sinir ağlarının ateroskleroz’un tahmin edilmesinde oldukça yararlı olacağı sonucuna varılabilir. Ancak örnek sayısının az olması göz önünde bulundurulduğunda, daha güvenilir sonuçlar elde edebilmek için örnek sayısının artırılması önerilebilir[5].

2012 yılında E.Saraç tarafından gerçekleştirilen bu çalışmada konutların değerlemesi için bir yapay sinir ağları modeli geliştirilmiştir. Modelin geliştirilmesi aşamasında Sermaye Piyasası Kurulu (SPK) ve Bankacılık Düzenleme ve Denetleme Kurumu (BDDK) lisanslı bir gayrimenkul değerleme firmasının İstanbul ilinin farklı ilçelerinde konumlu toplam 400 değerleme raporu analiz edilmiştir. Raporlardan gayrimenkulün değerini etkileyen 12 parametre seçilerek sayısallaştırılmıştır. Sayısallaştırılan veriler ile yapay sinir ağı oluşturulup 28 farklı model denenmiştir. Modellerin başarı oranları değişken olup Çok Katmanlı Algılayıcı (MLP) modeli ile %87 doğruluk oranına ulaşmıştır[9].

**2.Materyal ve Yöntem**

**2.1.Kullanılan Veritabanı**

Bu Uygulamada kullanılan veri seti UCI Machine Learning Repository’ den alınan “Tic-Tac-Toe Endgame” veritabanıdır. Maryland’ daki Johns Hopkins Üniversitesi’nden David W. Aha tarafından oluşturulmuştur. Kullanılan veritabanında 958 örneğe ait olan veri ve bu verilerin sonuçları olarak 2 adet sınıf bulunmaktadır. Bu sınıflar positive ve negative olarak ayrılmaktadır. Veritabanında kullanılan bu örneklerin 626 adeti positive(pozitif) sınıfına aitken 332 adet örnek ise negative(negatif) sınıfına aittir. Kullanılan bu veri setinde toplam 9 adet öznitelik kullanılmaktadır. Bu öznitelikler ve değerleri Tablo-1 de gösterilmektedir.

|  |  |  |
| --- | --- | --- |
|  | **Veri Seti** | **Değerler** |
| **1** | Top-Left-Square | X, O, B(Blank) |
| **2** | Top-Middle-Square | X, O, B(Blank) |
| **3** | Top-Right-Square | X, O, B(Blank) |
| **4** | Middle-Left-Square | X, O, B(Blank) |
| **5** | Middle-Middle-Square | X, O, B(Blank) |
| **6** | Middle-Right-Square | X, O, B(Blank) |
| **7** | Bottom-Left-Square | X, O, B(Blank) |
| **8** | Bottom-Middle-Square | X, O, B(Blank) |
| **9** | Bottom-Right-Square | X, O, B(Blank) |
| **10** | Class | Positive, Negative |

**Tablo-1.**Veri Setleri ve Değer Aralıkları

Veri setinde bulunan örneklerin öznitelikler; “Top-Left-Square” üst sol kare, “Top-Middle-Square” üst orta kare, “Top-Right-Square” üst sağ kare, “Middle-Left-Square” orta sol kare, “Middle-Middle-Square” ortadaki kare, “Middle-Right-Square” orta sağ kare şeklinde adlandırılmaktadır. Bu kareler “x”, ”o” veya “blank(boş)” şeklinde değerler almaktadır. Veri setinde tahminde kullanılacak olan veriler 1-9 arasındaki özniteliklerdi. Tablodaki 11 numaralı öznitelik sınıf bilgisini içermektedir. Veri setinde yer alan 958 verinin 450 tanesi eğitim için ve 508 tanesi ise test için ayrılmıştır. Eğitim verisinin 288 tane örnek positive olarak, kalan 162 veri ise negative olarak sınıflandırılmıştır. Test verisinde ise 388 adet örnek positive, 170 adet veri ise negative olarak sınıflandırılmaktadır. Veri setinde seçtiğimiz eğitim örnekleri Kullanılan bu veri setinde yar alan değerlerden “x” olanlar 1 olarak, “o” olanlar 0 olarak ve “b” değerleri ise 0,5 olarak değiştirildi.

**2.2.Yapılan İş**

Veritabanında bulunan örneklerimiz, girilen 9 özniteliğin sonucunda XOX oyununda x’ in kazanma durumunu bulmaya çalışmaktadır. Bu sebepten dolayı 9 adet giriş nöronu kullanılmıştır. İlk olarak UCI Machine Learning’ den alınan “Tic-Tac-Toe Endgame” veri seti Microsoft Excel kullanılarak, “Orijinal. xlsx” olarak kaydedilmiştir. Daha sonra kaydedilen bu veri setinde yer alan örneklerin özniteliklerinden “x” olan veriler 1 olarak, “o” olan veriler 0 olarak ve “b” olan veriler ise 0,5 olarak değiştirilmiştir. Veri setimizde kullanılan sınıflarımızda ise “positive” olan veriler 1 olarak “negative” olan veriler ise 0 olarak değiştirilmiştir. Yani veri setimizdeki öznitelikler, XOX oyununda yer alan 3x3 lük oyun alanında yatay olarak, ilk 3 özniteliğimiz, oyun alanımızın üstte yer alan 3 kareyi temsil etmektedir. Ardından gelen 3 öznitelik, ortada yer alan 3 kareyi, daha sonraki 3 öznitelik ise oyun alanının alt kısmında yer alan 3 kareyi temsil etmektedir. Veri setinin çıkışlarını niteleyen sınıflar ise oyundaki x karakterinin özniteliklere göre kazanma durumunu incelemektedir. Daha sonra orijinal veri olarak kaydettiğimiz veri setinde yer alan örneklerden 450 adetin öznitelikleri, oluşturulan “EgitimVeri. xlsx” isminde oluşturulan excel dosyasına aktarılmıştır. Daha sonradan bu örnekler kullanılarak ağın eğitilmesi amaçlanmıştır. Ardından bu 450 adet örneğin sonuçları ise “EgitimTarget. xlsx” isimli excel dosyasına aktarılmıştır. Bu örneklerimiz ise ağ eğitilirken, her örnek için ulaşılması gereken sonuçlar olarak belirlenmiştir. Ardından orijinal veride kalan 508 veri ise test için kullanılmak üzere “Test. xlsx” ve “TestTarget. xlsx” adlarındaki excel dosyalarına aktarılmıştır. Bu veriler ise ağın eğitimi sonucunda bulunan ağırlık değerleri ve bias değerleri kullanılarak, ağın öğrenme durumunu kontrol etmek için kullanılmıştır. Sonuç olarak da veri setinde yer alan örneklerin sonucu ya pozitif olabilir ya da negatif olabilir. Bu sebepten dolayı yapay sinir ağımızda 1 adet çıkış nöronumuz bulunmaktadır. Veri setimizde bulunan örneklerimizin öznitelik değerleri 0,1 ve 0,5 olduğundan normalizasyon yapılmasına gerek duyulmamıştır.

Seçtiğimiz bu veri setine uygun bir yapar sinir ağı oluşturmak için oluşturulan algoritmanın kodlanma işlemi için Matlab programı seçilmiştir. Matlab programının seçilme amacı ise yapay sinir ağlarında sıklıkla kullanılan martis çarpımlarını daha kolay bir biçimde yapabilmektir. Aynı şekilde Matlab üzerinde matematik işlemleri daha kolay bir biçimde ifade edilebilmektedir.

Yapay sinir ağımızda kullandığımız aktivasyon fonksiyonumuzu ise kolay türevlenebilir olması sebebi ile sigmoid fonksiyonu seçilmiştir. Başlangıçta sabit ağırlık ve bias değerleri oluşturulmuş ve eğitim boyunca bu ağırlıklar ve bias güncellenerek son değerlere ulaşılmıştır. Çıkış nöronunda oluşan hatanın geri yayılımı sayesinde tahmindeki hata oranı en aza indirilmeye çalışılmıştır.

**Formül 2.2.1.**Aktivasyon Fonksiyonu (Sigmoid)

Daha sonraki adım olarak verilerimizi Matlab üzerinde bir takım işlemlere tabi tutarak ağımızın özniteliklere göre doğru çıkışlar üretmesini sağlamak üzere kodlama işlemi yapılmaya başlanmıştır. Ağımızın ara katmanlarının çıkışlarının hesaplanması için aşağıdaki “On” formülü kullanılmıştır. Bu formül üzerinde “On” ara katmanlarımızın çıkışı, “Win” ve “Bn” ise ara katmanlarımızın ağırlıkları ve biasları olarak ifade edilmektedir.

**Formül 2.2.2.**Ara Katman Çıkış Formülü

Ardından yapay sinir ağımızda bulunan çıkışlarımızın, kullandığımız örneklerin özniteliklerine göre sonuçlarını bulmak için aşağıda yer alan “Çıkış Katmanı Sonuç Formülü” kullanılmaktadır. Bu formül üzerinde yer alan “Yn” çıkış katmanımızın değerlere göre sonuçlarını ifade etmektedir. “W’in” ve “B’n” ifadelerimiz ise çıkış katmanı ağırlıklarımızı ve biaslarımızı tanımlayabilmek için kullanılmaktadır.

**Formül 2.2.3.**Çıkış Katmanı Sonuç Formülü

Daha sonra kod içerisinde oluşturduğumuz döngü sayesinde eğitim sonucunda bulunan ağırlık ve biaslarımız kullanılarak veri setimizde yer alan test verilerimizin sonuçlarının doğruluk oranını göz önünde bulundurularak, en iyi sonucu veren “Mü” değeri ve “Alfa” değeri bulunmaya çalışılmıştır.

**Şekil 2.2.4.**Ara Katman Nöronlarına Göre Doğruluk Oranları

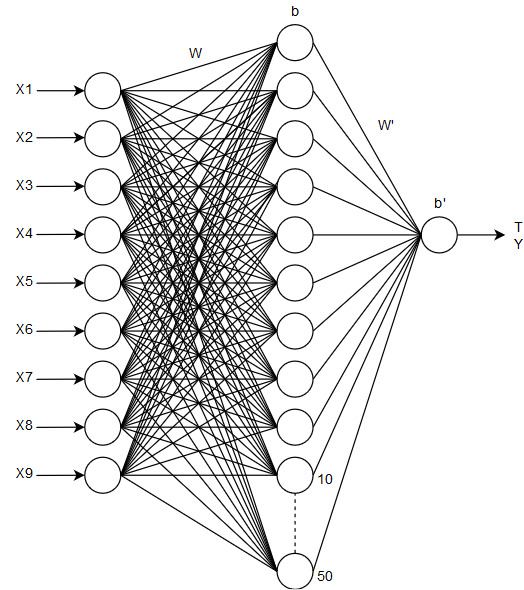
Şekil 2.2.4 de görüldüğü üzere döngü sayesinde 0 ile 1 arasındaki her bir alfa ve mü için en iyi doğruluk değerleri bulunmuştur. Grafik sonucunda en iyi sonucu veren alfa değerimiz 82.6771 doğruluk oranı ile 0,9 bulunmuştur. En iyi mü değerimiz ise 82,6771 doğruluk oranı ile 0,2 bulunmuştur.

Kullandığımız veri setindeki sınıflarımız göz önünde bulundurularak, oluşturduğumuz yapay sinir ağımızın çıkış katmanı nöronu olarak tek nöron kullanılmaktadır. Çünkü tek nöronla 2^1=2 adet sınıfı niteleyebilmekteyiz. Kullandığımız veri setinde 9 adet öz nitelik bulunduğundan yapay sinir ağımızda 9 adet giriş nöronu kullanılmaktadır. Son olarak ağımızda kullanacağımız ara katman nöron sayısını belirlemek için tekrar kod içerisindeki döngü kullanılarak en iyi doğruluk değerini bize veren ara katman nöron sayısı belirlenmeye çalışılmıştır.

**Şekil 2.2.5.**Ara Katman Nöronlarına Göre Doğruluk Oranları

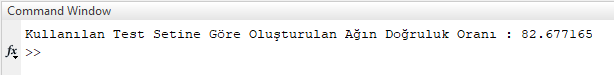
Döngü sayesinde her ara katman nöronu için elde edilen doğruluk değeri bulunmuştur. Şekil 2.2.5 de görüleceği üzere alfa ve mü değerleri de göz önünde bulundurularak ara katman nöronlarına göre doğruluk oranları belirlenmiştir. Grafikten anlaşılacağı üzere oluşturulan yapay sinir ağında kullanılması planlanan ara katman nöron sayısı 50 olarak seçilirse en yüksek doğruluk değeri olan 82.6 ya ulaşmaktadır. Ara katman nöron sayısı 50 den sonra azalmaktadır.

Kullanılan veri setine göre oluşturulan yapay sinir ağımızın gerekli nöronlarının sayısı belirlendikten sonra, ağımızın grafiğiniz çizimi Şekil 2.2.6 deki gibi oluşmaktadır. Görüldüğü gibi ağda 9 adet giriş nöronu, 50 adet aran nöron, 1 adet çıkış nöronu kullanılmıştır. Girişlerimiz X1 den X9’a kadardır.



**Şekil 2.2.6.**Yapay Sinir Ağı Grafiği

Oluşturduğumuz algoritmaya göre formüller kullanılarak kodlama işlemi Matlab üzerinde yapılmıştır. Algoritmaya göre kodlama işlemleri gerçekleştirilirken hiçbir şekilde hazır fonksiyonlar kullanılmamıştır. Matlab üzerinde oluşturulan kodlama kısmı ise ek kısmında yer verilmiştir. Son olarak ara nöron sayısı belirlenip ağımızı eğitime soktuktan sonra bulunan ağırlık ve biaslar kullanılarak test veri seti kullanılarak algoritmamızın doğruluk oranı hesaplanmıştır. Doğruluk oranımız ise Şekil 2.2.7 da gösterilmiştir.



**Şekil 2.2.7.**Doğruluk Oranı

Oluşturduğumuz yapay sinir ağımız eğitilirken kullanılan güncelleme formülleri aşağıdaki gibidir.

i=(Ti-Yj)\*Yj\*(1-Yj)

ΔW’ij(t+1)= μ\*j\*Oi+alfa\* ΔW’ij(t) ,j=1 ve i=1,…,50.

=+ ΔW’ij(t+1)

ΔWij(t+1)= μ\*Oi(1-Oj)\*1\*Wij\*Xi, j=1,…50 ve i=1,…,9.

=+ ΔWij(t+1)

Δb’ij(t+1)= μ\*j+alfa\* Δb’ij(t), i=1.

=+ Δb’ij(t+1)

Δbij(t+1)= μ\*Oi(1-Oi)\*1\*Wji, i=1,…,50 ve j=1,…,9.

=+ Δbij(t+1)

Formüllerde kullandığımız “t” ile ifade ettiğimiz değerimiz hedeflenen çıkış değerimizi ifade etmektedir.

**3.Sonuçlar ve Tartışma**

Yapılan bu çalışmada, UCI Machine Learning’ den elde edilen veri seti kullanılarak oluşturulan yapay sinir ağları ile XOX oyununda, her bir örnek için verilen göz önünde bulundurulan öznitelikler kullanılarak X karakterinin kazanma durumunu tahmin etme mekanizması oluşturulmuştur. Veri tabanındaki birbirinden farklı her bir örneğe ait öznitelik değerleri, oluşturulan yapay sinir ağının girişlerine verilmiş ve belli bir oranda hata oranı elde edilmiştir. Bu hata oranı algoritmada belirlediğimiz belli bir seviyenin altına inene kadar geri besleme yapılarak eğitim veri seti ile ağ eğitilmiştir. Eğitim sırasında 958 adet örneğin yalnızca 450 tanesi kullanılmıştır. Geri kalan 508 adet örnek ise eğitilen ağı test etmek için kullanılmıştır. Eğitimde kullanılan 288 adet örnek “positive” sınıfına aitken 162 örnek ise “negative” sınıfına aittir. Testte kullanılan örneklerin 338 tanesi “positive” ve 170 tanesi ise “negative” sınıfına ait verilerdir. Ağın başarısı çoğu zaman %78’in üzerinde çıkmaktadır.

**4.Kaynaklar**

[1] Fausett L. , “Fundamentals of neural networks: architectures, algorithms and applications”, 1st Edition, Prentice-Hall International. (1994).

[2] Mimaroğlu S. ,“Artificial Intelligent Based Automatic Car Valuation System”, ELECO '2012 Elektrik - Elektronik ve Bilgisayar Mühendisliği Dergisi, (2012)

### [3] David B. Fogel, “Using evolutionary programming to create neural networks that are capable of playing tic-tac-toe”, International Conference on Neural Networks. (1993)

### [4] M Çuhadar - C Kayacan, “Yapay Sinir Ağları Kullanılarak Konaklama İşletmelerinde Doluluk Oranı Tahmini: Türkiye’deki Konaklama İşletmeleri Üzerine Bir Deneme”, Anatolia: Turizm Araştırmaları Dergisi, Cilt 16, Sayı 1, Bahar: 24-30. (2005)

### [5] Cemil Çolak, M. Cengiz Çolak, M. Ali Atıcı, “Ateroskleroz'un tahmini için bir yapay sinir ağı”, “Ankara Üniversitesi Tıp Fakültesi Mecmuası 2005; 58:159-162”. (2005)

[6] M. Z. Yılmazoğlu, “Bir Tekstil Fabrikasında Güneş Destekli Çift Etkili Bir Absorpsiyonlu Soğutma Sisteminin Uygulamasının Teknik ve Ekonomik Analizi”, Gazi Üniversitesi, Fen Bilimleri Enstitüsü, Yüksek Lisans Tezi, 2006, Ankara.

[7] U. Akyol, “İplik Bobininin Kurutulmasının Teorik İncelenmesi”, Trakya Üniversitesi, Doktora Tezi, Fen Bilimleri Enstitüsü, 2007, Edirne.

[8] Yayar R. “Otomobil Sahipliğini Etkileyen Faktörlerin Belirlenmesi Yönetim ve Ekonomi Dergisi 22. Cilt. (2015)

[9] Erhan Saraç, “Yapay Sinir Ağları Metodu ile Gayrimenkul Değerleme”, T.C. İstanbul Kültür Üniversitesi Fen Bilimleri Enstitüsü (2012)

**Ekler**

**1.Aktivasyon Fonksiyonu (f. m) Kodu**

function [y]=f (x)

for i=1:length(x)

y(i)=1/(1+exp(-x(i)));

end

end

**1.Eğitim ve Test (tic\_tac\_toe\_endGame. m) Kodu**

clc

sayac = 0;

aranoron = 10;

dogru\_say=0;

yanlis\_say=0;

nu = 0.2; % Öğrenme Katsayısı

alfa = 0.9; % Momentum Katsayısı

% dizi=zeros(100,4);

%

% say;

%

% for aranoron=1:1:55

% for nu=0.1:0.1:0.9

% for alfa=0.1:0.1:0.9

% say=say+1;

x = Egitim; % Girişler

w = zeros(aranoron,9); % Ara Katman Ağırlıkları

dw = zeros(aranoron,9); % Ara Katman Ağırlıklarındaki Değişim

b = zeros(aranoron,1); % Ara Katman Biasları

db = zeros(aranoron,1); % Ara Katman Ağırlıklarındaki Değişim

o = zeros(aranoron,1); % Ara Katman Çıkışları

wp = zeros(1,aranoron); % Çıkış Katmanları Ağırlıkları

dwp = zeros(1,aranoron); % Çıkış Katmanları Ağırlıklarındaki Değişim

bp = zeros(1,1); % Çıkış Katmanı Biasları

dbp = zeros(1,1); % Çıkış Katmanı Biaslarındaki Değişim

rp = zeros(1,1); % Çıkış Katmanı Hata Faktörleri

y = zeros(1,1); % Üretilen Çıkışlar

t = EgitimTarget; % Hedef Çıkışlar

%veriler üzerinde dolaşmak için for

for k=1:length(Egitim)

hata=1;

while(hata>0.01)

%ara katman çıkışlarının bulunması

for i=1:aranoron

top=0;

for n=1:9

top=top+x(n,k)\*w(i,n);

end

o(i)=f(top+b(i));

end

%çıkış katmanı çıkışlarının bulunması

for n=1:1

top=0;

for i=1:aranoron

top=top+o(i)\*wp(n,i);

end

y(n)=f(top+bp(n));

end

%%%%%%%%%%%%%%%%%%ÇIKIŞ KATMANI GÜNCELLEMELERİ%%%%%%%%%%%%%%

%çıkış hata faktörlerinin hesaplanması

for i=1:1

rp(i)=(t(1,k)-y(i))\*y(i)\*(1-y(i));

end

%çıkış ağırlıklarının güncellemesi

for j=1:aranoron

for i=1:1

dwp(i,j)=nu\*rp(i)\*o(j)+alfa\*dwp(i,j);

end

end

%çıkış katmanı biaslarının güncellenmesi

for j=1:1

dbp(j)=nu\*rp(j)+alfa\*dbp(j);

bp(j)=bp(j)+dbp(j);

end

%%%%%%%%%%%%%% ARA KATMAN GÜNCELLEME %%%%%%%%%%%%%%%%%%

%ara katman agırlıkları güncellemesi

for i=1:aranoron

for j=1:9

dw(i,j)=nu\*o(i)\*(1-o(i))\*wp(1,i)\*rp(1)\*x(j,k);

w(i,j)=w(i,j)+dw(i,j);

end

end

%ara katman biaslarının güncellenmesi

for i=1:aranoron

db(i)=nu\*o(i)\*(1-o(i))\*wp(1,i)\*rp(1);

b(i)=b(i)+db(i);

end

wp=wp+dwp;

sayac=sayac+1;

hata=(1/2)\*((t(1,k)-y(1))^2);

end%while sonu

end

sonuc = zeros(2,length(Test));

%%%%%%%%%%%% Test Aşaması %%%%%%%%%%%%

dogru=0;

for k=1:length(Test)

%ara katman çıkışlarının bulunması

for i=1:aranoron

top=0;

for n=1:9

top=top+Test(n,k)\*w(i,n);

end

o(i)=f(top+b(i));

end

%çıkış katmanı çıkışlarının bulunması

for n=1:1

top=0;

for i=1:aranoron

top=top+o(i)\*wp(n,i);

end

y(n)=f(top+bp(n));

end

for i=1:1

if y(i)>0.5

y(i)=1;

else

y(i)=0;

end

end

sonuc(1,k) = y(i);

sonuc(2,k) = TestTarget(1,k);

if(TestTarget(1,k)==y(1))

dogru\_say=dogru\_say+1;

dogru=dogru+1;

else

yanlis\_say=yanlis\_say+1;

end

end%for sonu

dogruluk = 100\*dogru/length(Test);

% dizi(say,1)=aranoron;

% dizi(say,2)=nu;

% dizi(say,3)=alfa;

% dizi(say,4)=(100\*dogru/length(Test));

fprintf('Kullanılan Test Setine Göre Oluşturulan Ağın Doğruluk Oranı : %f\n',dogruluk);

% end

% end

% end