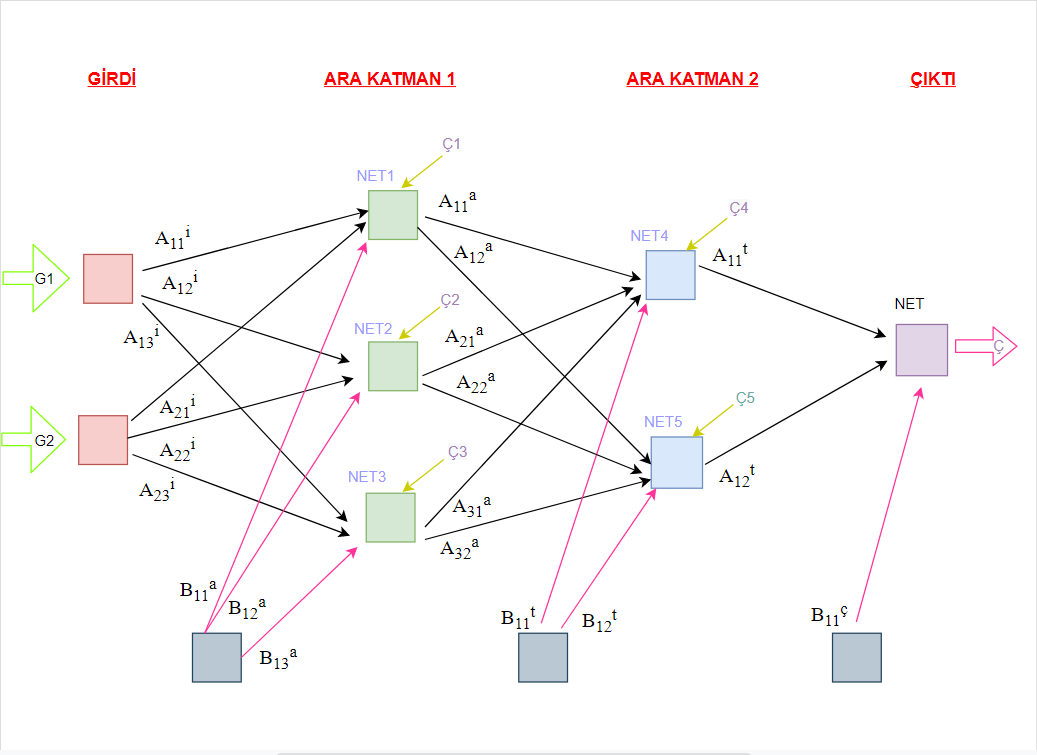
ÇOK KATMANLI YAPAY SİNİR AĞI



Çok katmanlı yapay sinir ağları; girdi katmanı, ara katmanlar ve çıktı katmanlarından oluşmaktadır. Çok katmanlı yapay sinir ağları karmaşık problemlerin çözümünde kullanılmaktadır. Dolayısıyla doğrusal olmayan problemlerde tercih edilmektedir. Birden çok girdiye ve birden fazla ara katmana sahip olabilmektedir. Ara katmanların sayısı problemin akışına göre arttırılabilir veya azaltılabilir.Ara katman problemin yapısına göre farklı fonksiyonlar ile işlenip çıktı katmanına aktarılmasını sağlamaktadır.

Çok katmanlı ağlarda AND probleminin farklı momentum öğrenme katsayısı ve başlangıç ağırlık değerleri çözümü ;

**AND**

Girişler Çıkış

0 0 0

0 1 0

1 0 0

1 1 1

AND probleminde girişlerden herhangi biri 0 olursa çıkış da 0 olur.Diğer durumda ise 1 olur.

Verilen momentum,öğrenme katsayısı ve başlangıç değerleri kullanılarak ;2 girişli ,birinci ara katmanı 3 proses elemanından oluşan ikinci ara katmanı ise 2 proses elemanından oluşan ,1 çıkışlı çok katmanlı bir ağ tasarlayacağız.

Giriş Katmanı: Sisteme giriş olarak gelen veriler bu katmanda yer alır. Bu katmanda giriş verileri üzerinde hiçbir değişiklik yapmadan bir sonraki katman olan ara katmana aktarır.

Ara Katman: Verinin transfer edildiği katmandır. Öğrenme bu katmanda olur.

Çıkış Katmanı: Sistemin giriş verilerine göre öğrenmesini istenilen çıkış değerleri burada yer alır. Sistem çıktısının alındığı yerdir.

Aktivasyon Fonksiyonu: Nörona gelen bilginin bir sonraki nörona iletilip iletilmeyeceğine karar veren birimdir.

Eşik Değeri : Sistem performansını etkiler . Bias değeri, aktivasyon fonksiyonunu sağa veya sola ötelenmesini sağlar.

* Öğrenme katsayısı ( λ)= 0.9
* Momentum(a) =0.1

AĞIRLIK DEĞERLERİ

* Girdi katmanı ile ara katman arasındaki başlangıç ağırlıkları = Ai

Ai = [ [0.131 0.443 0.624]

[-0.347 -0.445 -0.127] ]

Ai =  [[ A11i A12i A13i  ]

[ A21i A22i A23i  ]]

* 1. Ara katman ile 2. Katman arasındaki ağırlık değerleri:

Aa = [ [0.978 0.956]

[0.724 -0.932]

[-0.946 0.923] ]

Aa = [ [A11a A12a]

[A21a A22a]

[A31a A32a] ]

* 2.Ara katman ile çıktı arasindaki başlangıç agırlıkları

At = [0.987 0.966]T = [A11t A12t]

EŞİK DEĞERLERİ

* Giriş katmanı ile 1. katman arasındaki eşik değeri:

Ba = [0.526 0.513 0.552]= [B11a B12a  B13a]

* 1. Katman ile 2. katman arasındaki eşik değeri

Bt = [0.524 0.525]= [B11t B12t]

* 2. Katman ile çıkış katmanı arasındaki eşik değeri

Bç = [0.574]= [B11ç]

AKTİVASYON FONSKSİYONU

* Ara katman aktivasyon fonksiyonu

Net >= 0 ise 1

Net < 0 ise 0

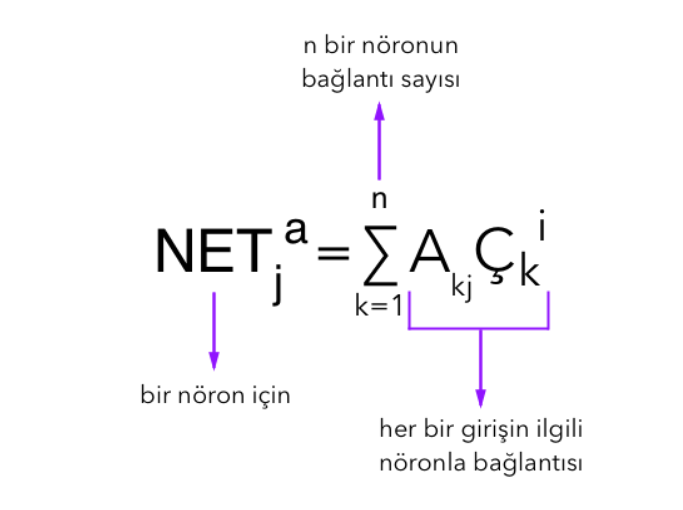
* Çıkış aktivasyon fonksiyonu

F(NET) = 1 / (1 + e-NET  )

İLERİ YAYILIM

Bir eğitim kümesindeki girdilerin belirlenen parametreler ile yapay sinir ağında soldan sağa doğru ilerleyen işlemlere denir. Yani girdinin yapay sinir ağı modeline girmesinden tahminin yapılmasına kadar olan süreçtir.

* Tek yönlü bilgi akışı söz konusudur.
* Bu ağ modelinde Girdi tabakasından alınan bilgiler ara katmana iletilir.



1. Giriş ile 1. Ara Katman İleri Yayılım

G1 = 0, G2 = 0, Ç = 0

Neti = GT \* Ai + Bt + T \* +

Neti =

* Net1 = Aİ11G1 + Aİ21G2 + Ba11

= (0.131 \* 0) + (-0.347 \* 0) + 0.526 = 0.526

Net1 ≥ 0 ise 1

Ç1 = 1

* Net2 = Aİ12G1 + Aİ22G2 + Ba12

= (0.443 \* 0) + (-0.445 \* 0) + 0.513 = 0.513

Net2 ≥ 0 ise 1

Ç2 = 1

* Net3 = Aİ13G1 + Aİ23G2 + Ba13

= (0.624 \* 0) + (-0.127 \* 0) + 0.552 = 0.552

Net3 ≥ 0 ise 1

Ç3 = 1

2. 1. Ara Katman ile 2. Ara Katman Arasındaki İleri Yayılım

* Net4 = Aa11Ç1 + Aa21Ç2 + Aa31 Ç3 +Bt11

= (0.978 \* 1) + (0.724 \* 1) -(0.946 \* 1 ) +0.524 = 1.28

Net4 ≥ 0 ise 1

Ç4 = 1

* Net5 = Aa12Ç1 + Aa22Ç2 + Aa32 Ç3 +Bt12

= (0.956 \* 1) + (-0.932 \* 1) +(0.923 \* 1 ) + 0.552 = 1.472

Net5 ≥ 0 ise 1

Ç5 = 1

* Net= At11 Ç4+ At21 Ç5+ BÇ11

= (0.987\* 1) +( 0.966\* 1) + 0.574 = 2.527

Ç’ = = 0.926

İleri yayılım ile Net1 , Ç1 ; Net2, Ç2 ; Net3 , Ç3 ; Net4 , Ç4 ; Net5 , Ç5 ve NetS , Ç değerleri bulunur.Eğer B=Ç ise diğer girişler için ileri yayılım yapılır.Eğer B≠Ç ise geri yayılım yaparak yeni Aİ , Aa ,At ; Ba, Bt , Bç değerleri bulunur.

B≠Ç ise Ağın Hatası (E) = B - Ç

E = B - Ç = 0 – 0.926 = - 0.926

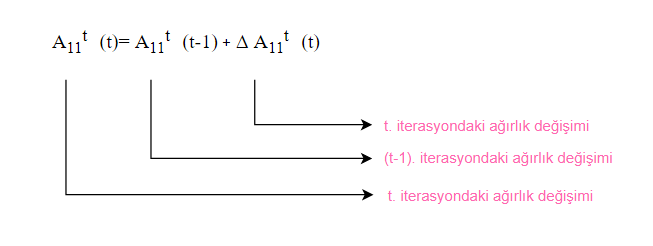
Çıktı Ünitesi Hatası

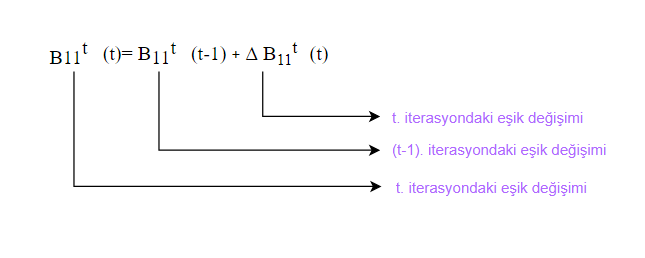
δ= f’(NET) \* E => Ç \* (1 – Ç ) \* E

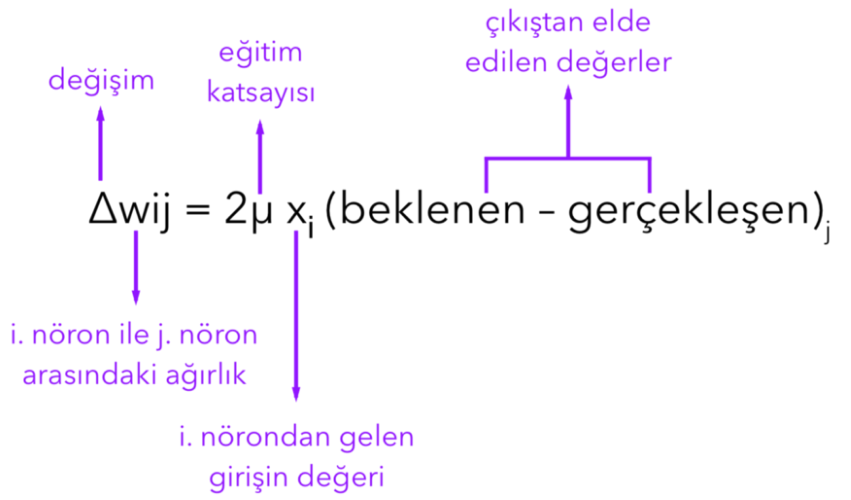
δ= 0.926 \* (1 – 0.926) \*(-0.926 ) = -0.0634

GERİ YAYILIM

İleri beslemeli ağ ile elde edilen çıktı değerleri ile gerçek sistem çıktısı arasındaki hata oranlarına bakılarak hatanın geriye doğru yayılımı ile giriş katmanına kadar iletilmesi işlemi için geçen süreçtir.Bu çeşit yapay sinir ağların dinamik hafızaları vardır ve bir andaki çıkış hem o andaki hem de önceki girişleri yansıtır.







1.Çıkış ile 2. Katman arası geri yayılımı

At11(t) = At11(t-1) +ΔAt11(t)

At11(t)= t. İterasyondaki ağırlık değeri

At11(t-1) = (t-1). İterasyondaki ağırlık değeri

ΔAt11(t)= t. İterasyondaki ağırlık değişimi

Bt11(t) = Bt11(t-1) +ΔBt11(t)

Bt11(t)= t. İterasyondaki eşik değeri

Bt11(t-1) = (t-1). İterasyondaki eşik değeri

ΔBt11(t)= t. İterasyondaki eşik değişimi

Yeni At değerleri hesaplama

ΔAt11(t)= ᵩ x δ x Ç4 + a ΔAt11(t-1)= (0.9 x -0.0634 x 1) + 0.1 x 0 = -0.05706

At11(t) = At11(t-1) +ΔAt11(t) = 0.987-0.05706= -0.9299

ΔAt21(t)= ᵩ x δ x Ç5 + a ΔAt21(t-1)= (0.9 x -0.0634 x 1) + 0.1 x 0 = -0.05706

At21(t) = At21(t-1) +ΔAt21(t) = 0.966-0.05706= -0.9089

Yeni At değerleri:

At = [-0.9299 0.9089]T

Yeni Bç değerleri hesaplama

Bç(t) = Bç (t-1) +Δ Bç (t) = 0.987-0.05706= -0.9299

Δ Bç (t)= ᵩ x δ + a Bç (t-1)= (0.9 x -0.0634) + 0.1 x 0 = -0.05706

Bç (t) = 0.574- 0.05706 = 0.5109

Yeni Bç değeri

Bç11 =[0.5169]

2. Katman ile 1. Katman arası geri yayılım

Aa değerleri

ΔAa11(t)= ᵩ x δ x Ç1 + a ΔAa11(t-1)= (0.9 x -0.0634 x 1) + 0.1 x 0 = -0.05706

Aa11(t) = Aa11(t-1) +ΔAa11(t) = 0.978-0.05706= 0.9209

ΔAa12(t)= ᵩ x δ x Ç1 + a ΔAa12(t-1)= (0.9 x -0.0634 x 1) + 0.1 x 0 = -0.05706

Aa12(t) = Aa12(t-1) +ΔAa12(t) = 0.956-0.05706= 0.8989

ΔAa21(t)= ᵩ x δ x Ç2 + a ΔAa21(t-1)= (0.9 x -0.0634 x 1) + 0.1 x 0 = -0.05706

Aa21(t) = Aa21(t-1) +ΔAa21(t) = 0.124-0.05706= 0.6669

ΔAa22(t)= ᵩ x δ x Ç2 + a ΔAa22(t-1)= (0.9 x -0.0634 x 1) + 0.1 x 0 = -0.05706

Aa22(t) = Aa22(t-1) +ΔAa22(t) = 0.932-0.05706= -0.9891

ΔAa31(t)= ᵩ x δ x Ç3 + a ΔAa31(t-1)= (0.9 x -0.0634 x 1) + 0.1 x 0 = -0.05706

Aa31(t) = Aa31(t-1) +ΔAa31(t) = 0.946-0.05706= -1.0031

ΔAa32(t)= ᵩ x δ x Ç3 + a ΔAa32(t-1)= (0.9 x -0.0634 x 1) + 0.1 x 0 = -0.05706

Aa32(t) = Aa32(t-1) +ΔAa32(t) = 0.923-0.05706= 0.8659

Yeni Aa değerleri

Aa =

Bt değerleri

Δ Bt11 (t)= ᵩ x δ + a Bt11 (t-1)= (0.9 x -0.0634) + 0.1 x 0 = -0.05706

Bt11(t) = Bt11 (t-1) +Δ Bt11 (t) = 0.524-0.05706= 0.4669

Δ Bt12 (t)= ᵩ x δ + a Bt12 (t-1)= (0.9 x -0.0634) + 0.1 x 0 = -0.05706

Bt12(t) = Bt12 (t-1) +Δ Bt12 (t) = 0.525-0.05706= 0.4679

Yeni Bt  değerleri

Bt = [0.4669 0.4679]

**MATLAB KOD**

**GD = [ 0 0; 0 1; 1 0; 1 1]; %durumların tanitilmasi**

**GK\_A1 = [0.131 0.443 0.624; -0.347 -0.445 -0.127 ]; %Giriş katmanı ile ara katman arasındaki ağırlıklar**

**A1\_A2 = [0.978 0.956; 0.724 -0.932; -0.946 0.923 ]; %Birinci ara katman ile ikinci ara katman arasındaki ağırlıklar**

**A2\_C = [0.987 0.966] ; %İkinci ara katman ile çıkış katmanı arasındaki ağırlıklar**

**B1 = [0.526 0.513 0.552]; % I. esik degeri**

**B2 = [0.524 0.525]; % II. esik degeri**

**B3 = [0.574]; % III. esik degeri**

**bk = [0 0 0 1]; %beklenen durumlar**

**A2\_C\_O = [0 0] ; %A2\_C(t - 1)**

**B3\_C\_O = [0]; %B3\_C(t - 1)**

**A1\_A2\_O = [0 0;0 0;0 0] ; %A1\_A2(t - 1)**

**B2\_C\_O = [0 0]; %B2\_C(t - 1)**

**GK\_A1\_O = [0 0 0; 0 0 0] ; %A1\_C(t - 1)**

**B1\_C\_O = [0 0 0]; %B1\_C(t - 1)**

**fi = 0.9 ; %ogrenme katsayisi**

**mem = 0.1; %momentum**

**a = 1;**

**iterasyon = 0;**

**for sayici = 0 : 2**

**while a < 5**

**iterasyon = iterasyon + 1;**

**%ileri yayilim**

**net\_1 = GD(a, 1 : 2) \* GK\_A1 +B1**

**net\_1 = step\_fonk(net\_1)**

**net\_2 = (net\_1 \* A1\_A2) + B2**

**net\_2 = step\_fonk(net\_2)**

**net\_3 = (net\_2 \* (A2\_C)') + B3**

**net\_3 = sgm\_fonk(net\_3)**

**%hata kontrolu**

**E = bk(a) - net\_3**

**if(abs(E) < abs(0.01))**

**fprintf('hata yok \n');**

**a = a + 1;**

**else if(abs(E) >= abs(0.01) )**

**z = sgm\_trv\_fonk(net\_3);**

**sigma = z \* E**

**%geri yayilim ara\_katman\_2 - cikis**

**A2\_C\_Y = geri\_yayilim\_c(A2\_C,sigma,net\_2,A2\_C\_O)**

**A2\_C\_O = A2\_C;**

**A2\_C = A2\_C\_Y;**

**B3\_Y = geri\_yayilim\_bc(B3 ,sigma , B3\_C\_O)**

**B3\_C\_O = B3;**

**B3 = B3\_Y;**

**% ara\_katman\_1 - ara\_katman\_2**

**A1\_A2\_Y = geri\_yayilim\_c(A1\_A2, sigma, (net\_1)', A1\_A2\_O)**

**A1\_A2\_O = A1\_A2;**

**A1\_A2 = A1\_A2\_Y;**

**B2\_Y = geri\_yayilim\_bc(B2 ,sigma , B2\_C\_O)**

**B2\_C\_O = B2;**

**B2 = B2\_Y;**

**% ara\_katman\_1 - ara\_katman\_2**

**GK\_A1\_Y = geri\_yayilim\_c(GK\_A1, sigma, (GD(a, 1 : 2))',GK\_A1\_O)**

**GK\_A1\_O = GK\_A1;**

**GK\_A1 = GK\_A1\_Y;**

**B1\_Y = geri\_yayilim\_bc(B1 ,sigma , B1\_C\_O)**

**B1\_C\_O = B1;**

**B1 = B1\_Y;**

**a = 1;**

**end**

**end**

**end**

**a = 4;**

**GD(a, 1 : 2)**

**iterasyon**

**break**

**end**

Matlab Fonksiyonlar

**%\***

**%"Bt\_0 ------- Esik degerlerinin (t - 1)**

**%"Bt\_1 ------- Esik degerlerinin (t)**

**% "Bt\_2 ------- Esik degerlerinin (t + 1)**

**%\***

**function [y] = geri\_yayilim\_bc(Bt\_1 ,sigma , Bt\_0)**

**fi = 0.9 ; %ogrenme katsayisi**

**mem = 0.1; %momentum**

**Bt\_2 = Bt\_1 + (fi \* sigma + mem \* Bt\_0);**

**y = Bt\_2 ;**

**end**

**%\***

**%At\_0 ------- Agirlik degerlerinin (t - 1)**

**%At\_1 ------- Agirlik degerlerinin (t)**

**%At\_2 ------- Agirlik degerlerinin (t + 1)**

**%\***

**function [y] = geri\_yayilim\_c(At\_1,sigma,net,At\_0)**

**fi = 0.9 ; %ogrenme katsayisi**

**mem = 0.1; %momentum**

**At\_2 = At\_1 + (fi .\* sigma .\* net + mem .\* At\_0);**

**y = At\_2;**

**end**

**%\***

**%matris gelen matris degeri"**

**%\***

**function [y] = sgm\_fonk(matris)**

**[m,n] = size(matris);**

**%cikan degerler aktivasyon fonksiyonuna girer**

**for i = 1 : m**

**for j = 1 : n**

**matris(i,j) = 1 / (1 + exp (-matris(i,j)));**

**j = j + 1;**

**end**

**i = + 1;**

**end**

**y = matris;**

**end**

**%\***

**%matris gelen matris degeri"**

**%\***

**function [y] = sgm\_trv\_fonk(matris)**

**[m,n] = size(matris);**

**%cikan degerler aktivasyon fonksiyonuna girer**

**for i = 1 : m**

**for j = 1 : n**

**matris(i,j) = matris(i,j) \* (1 - matris(i,j));**

**j = j + 1;**

**end**

**i = + 1;**

**end**

**y = matris;**

**end**

**%\***

**%matris gelen matris degeri"**

**%\***

**function [y] = step\_fonk(matris)**

**[m,n] = size(matris);**

**%cikan degerler aktivasyon fonksiyonuna girer**

**for i = 1 : m**

**for j = 1 : n**

**if(matris(i,j) >= 0)**

**matris(i,j) = 1;**

**else**

**matris(i ,j) = 0;**

**end**

**j = j + 1;**

**end**

**i = + 1;**

**end**

**y = matris;**

**end**

**Python Kod**

"""GiriÅŸ katmanÄ± ile ara katman arasÄ±ndaki aÄŸÄ±rlÄ±klar

Birinci ara katman ile ikinci ara katman arasÄ±ndaki aÄŸÄ±rlÄ±klar

Ä°kinci ara katman ile Ã§Ä±kÄ±ÅŸ katmanÄ± arasÄ±ndaki aÄŸÄ±rlÄ±klar

I. esik degeri

II. esik degeri

III. esik degeri

beklenen durumlar

A2\_C(t - 1)

B3\_C(t - 1)

A1\_A2(t - 1)

B2\_C(t - 1)

A1\_C(t - 1)

B1\_C(t - 1)

ogrenme katsayisi"""

import numpy as np

import math as mt

"fonksiyon tanimlamalari"

"\*"

"matris gelen matris degeri"

"\*"

"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

def step\_fonk(matris):

(m,n) = matris.shape;

for i in range(m):

for j in range(n):

if(matris[i,j] >= 0):

matris[i,j] = 1;

else:

matris[i,j] = 0;

j = j + 1;

i = i +1;

return matris;

"""\*

"matris gelen matris degeri"

"\*

"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"""

def sgm\_fonk(matris):

m = len(matris);

for i in range(m):

matris[i] = 1 / (1 + mt.exp(-matris[i]));

i = i + 1;

return matris;

"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

def sgm\_trv\_fonk(matris):

m = len(matris);

for i in range(m):

matris[i] = matris[i] \* (1 - matris[i]);

i = i + 1;

return matris;

"""\*

"At\_0 ------- Agirlik degerlerinin (t - 1)

"At\_1 ------- Agirlik degerlerinin (t)

"At\_2 ------- Agirlik degerlerinin (t + 1)

"\*

"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"""

def geri\_yayilim\_c(At\_1,sigma,net,At\_0):

fi = 0.9;

mem = 0.1;

At\_2 = At\_1 + (fi \* sigma \* net + mem \* At\_0);

y = At\_2;

return y;

"""\*

"Bt\_0 ------- Esik degerlerinin (t - 1)

"Bt\_1 ------- Esik degerlerinin (t)

"Bt\_2 ------- Esik degerlerinin (t + 1)

"\*

"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"""

def geri\_yayilim\_bc(Bt\_1 ,sigma , Bt\_0):

fi = 0.9;

mem = 0.1;

Bt\_2 = Bt\_1 + (fi \* sigma + mem \* Bt\_0);

y = Bt\_2;

return y;

"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

"degisken tanimlamalari"

GD = np.array([[0.0, 0.0],

[0.0, 1.0],

[1.0 ,0.0],

[1.0 ,1.0]]);

GK\_A1 = np.array([[0.131 ,0.443, 0.624],

[-0.347 ,-0.445 ,-0.127]]);

A1\_A2 = np.array([[0.978, 0.956],

[0.724, -0.932],

[-0.946, 0.923 ]]);

A2\_C = np.array([0.987 ,0.966]);

B1 = np.array([0.526 ,0.513 ,0.552]);

B2 = np.array([0.524 ,0.525]);

B3 = np.array([0.574]);

bk = np.array([0.0 ,0.0, 0.0 ,1.0]);

A2\_C\_O = np.array([0 ,0]) ;

B3\_C\_O = np.array([0.0]);

A1\_A2\_O = np.array([[0,0],[0,0],[0,0]] );

B2\_C\_O = np.array([0 ,0]);

GK\_A1\_O = np.array([[0.0, 0.0, 0.0], [0.0 ,0.0 ,0.0]]) ;

B1\_C\_O = np.array([0.0, 0.0, 0.0]);

fi = 0.9 ;

mem = 0.1;

E = 0.0;

iterasyon = 0;

a = 0;

x = 0;

for x in range(1):

while a < 4 :

"ileri yayilim"

print("iterasyon", iterasyon);

iterasyon = iterasyon + 1;

net\_1 = np.dot(GD[a : a + 1, 0:2], GK\_A1[0 : 3,0: 4 ]) + B1;

print("net\_1", net\_1);

net\_1 = step\_fonk(net\_1);

print("net\_af", net\_1);

net\_2 = np.dot(net\_1, A1\_A2) + B2;

print("net\_2", net\_2);

net\_2 = step\_fonk(net\_2);

print("net\_2\_af", net\_2);

net\_3 = np.dot(net\_2, A2\_C.transpose()) + B3;

print("net\_3", net\_3);

net\_3 = sgm\_fonk(net\_3);

print("net\_3\_af", net\_3);

E = bk[a] - net\_3;

if(abs(E) < abs(0.01)):

print("hata yok \n");

a = a + 1;

elif(abs(E) >= abs(0.01)):

z = sgm\_trv\_fonk(net\_3);

sigma = z \* E;

A2\_C\_Y = geri\_yayilim\_c(A2\_C,sigma,net\_2,A2\_C\_O);

A2\_C\_O = A2\_C;

A2\_C = A2\_C\_Y;

print("a2\_c\_y \n");

print(A2\_C\_Y , A2\_C\_O ,A2\_C ,"\n");

B3\_Y = geri\_yayilim\_bc(B3 ,sigma , B3\_C\_O);

B3\_C\_O = B3;

B3 = B3\_Y;

print("b3\_y \n");

print(B3\_Y,B3\_C\_O, B3 ,"\n");

A1\_A2\_Y = geri\_yayilim\_c(A1\_A2,sigma, net\_1.transpose(),A1\_A2\_O );

A1\_A2\_O = A1\_A2;

A1\_A2 = A1\_A2\_Y;

print("GK\_A1\_Y \n");

print(A1\_A2\_Y ,"\n" , A1\_A2\_O ,"\n" ,A1\_A2 ,"\n" );

B2\_Y = geri\_yayilim\_bc(B2 ,sigma , B2\_C\_O);

B2\_C\_O = B2;

B2 = B2\_Y;

print("b2y \n");

print(B2\_Y, "\n" ,B2\_C\_O, "\n", B2,"\n");

GK\_A1\_Y = geri\_yayilim\_c(GK\_A1,sigma,GD[a : a + 1, ::].transpose(),GK\_A1\_O);

GK\_A1\_O = GK\_A1;

GK\_A1 = GK\_A1\_Y;

print("GK\_A1\_Y \n");

print(GK\_A1\_Y ,"\n" , GK\_A1\_O ,"\n" ,GK\_A1 ,"\n" );

B1\_Y = geri\_yayilim\_bc(B1 ,sigma , B1\_C\_O);

B1\_C\_O = B1;

B1 = B1\_Y;

print("b1y \n");

print(B1\_Y, "\n" ,B1\_C\_O, "\n", B1 ,"\n");

a = 0;

print("iterasyon =" , iterasyon);

Feyzanur Boztepe 171110015

Can Ahmet Acar 171110001

Tayfun Bacak 171110009

Ömer Faruk Çelik 161110020