SAKARYA
ÜNİVERSİTESİ

BSM 310
YAPAY ZEKA

CEMİL ÖZ, İSMAİL ÖZTEL

~ YAPAY SİNİR AĞLARI ~

KONULAR

- Yapay sinir ağlarına giriş
- Yapay sinir ağlarının kullanım alanları
- Yapay sinir ağlarında öğrenme
- Yapay sinir hücresi
- Perceptron (algılayıcı)
- Çok Katmanlı İleri Beslemeli Yapay Sinir Ağları
- Geri Yayılım Algoritması

Yapay sinir ağları - giriş

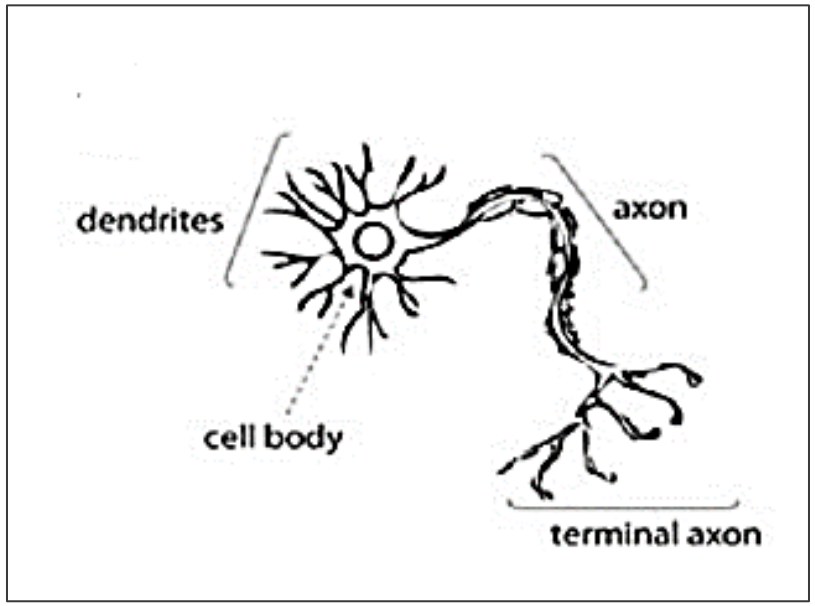
- Bilgisayarlar geniş uzayda tarama gerektiren konularda insanlardan daha hızlı ve güvenilir sonuçlar ortaya koymaktadır.
- Bilgisayarlar konuşulanı anlayıp, zeka oyunları oynayabiliyorsa bunun sebebi insan davranışlarının bilgisayara aktarılması değil; bilgisayarların güçlü hesaplama kabiliyetindendir.
- Bir makine doğal dil işleme, görüntü işleme gibi problemleri çözebiliyor olsa da, bu problemlerde insanın bilgisayara olan üstünlüğü açıktır.

Yapay sinir ağları - giriş

- İnsan beyni yaklaşık 10 milyar sinir hücresinden (nörondan) oluşur.
- Akıl almaz bir verimlilik ve paralellik ile çalışır. Örneğin, karmaşık görsel hesaplamalar 100ms den önce gerçekleştirilir.
- İnsan beyni, geleneksel bilgisayarlardan tamamen farklı bir yolla işlem yapar. Oldukça komplekstir, doğrusal değildir ve paralel dağıtık bir yapıya sahiptir.
- Yapay sinir ağları (YSA), insan beyninin çalışma sisteminin yapay olarak benzetimi çabalarının bir sonucu olarak ortaya çıkmıştır.

Biyolojik sinir hücresi

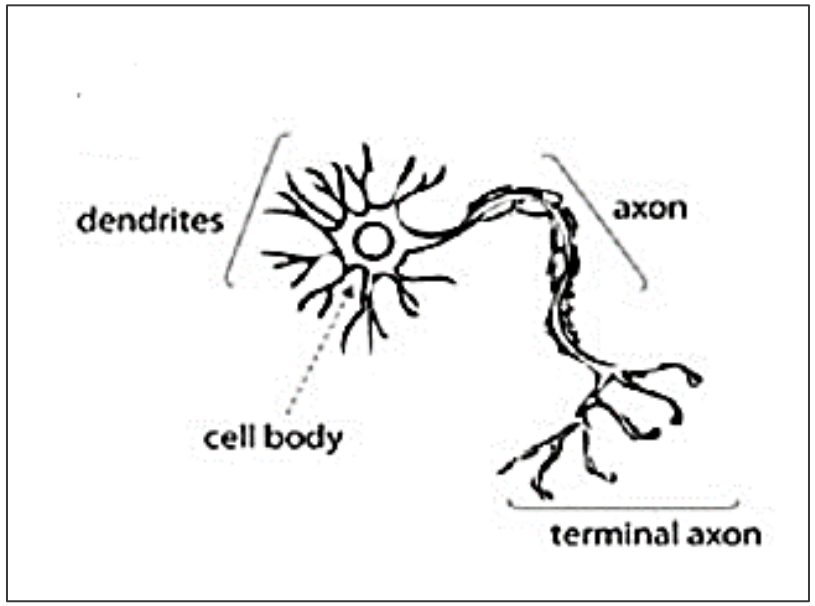
- Biyolojik sinir hücreleri temelde üç bölgeye ayrılır:
  - Soma: hücreyi denetler ve hücredeki etkinlikleri yönetir.
  - Dendrit: bilgiyi diğer hücrelerden almakla sorumludur.
  - Akson: gövdedeki bilgiyi diğer hücrelere iletmekle sorumludur.



https://www.researchgate.net/publication/335190001_New_neural_network_for_real-time_human_dynamic_motion_prediction/figures?lo=1

Biyolojik sinir hücresi

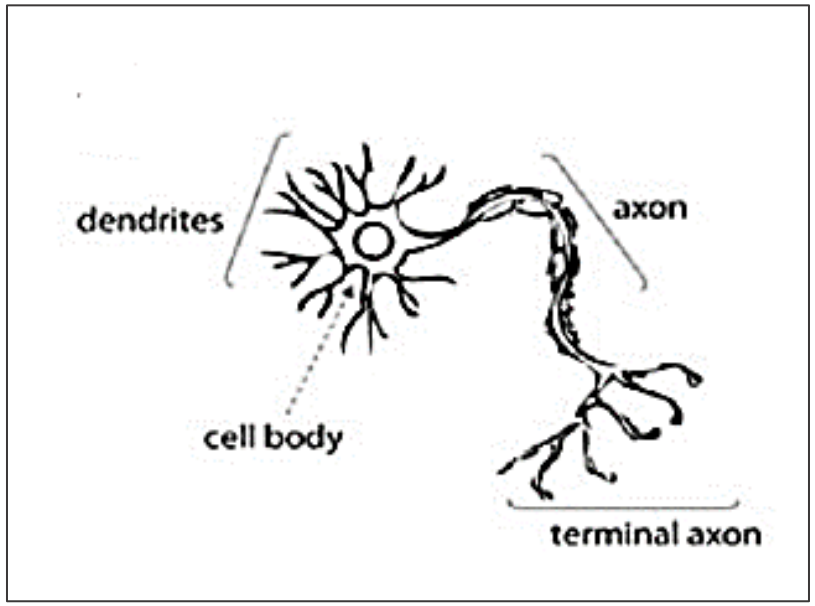
- Sinir hücreleri birbiri ile doğrudan temas halinde değildir.
- Dentritler ile akson uçları arasında boşluk vardır.
- Hücre gövdesinden iletilen sinyal akson boyunca terminal uçlara taşınır ve gelen sinyal ile birlikte nörotransmitter adı verilen kimyasal salgılanır.
- Bu kimyasal boşlukta yayılır ve bir sonraki sinir hücresinin dentriti tarafından sinyal alınır.



Biyolojik sinir hücresi

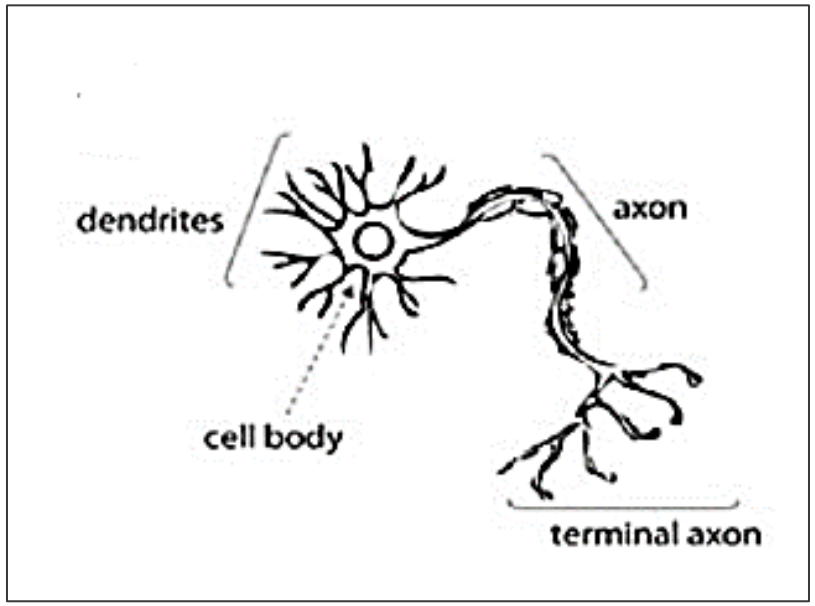
- Akson ucu terminaller ile dentritler arası sinaps boşlukları bilgilerin uzun süre saklandığı bilgi saklama birimi olarak düşünülmektedir.

- Uzun süreli saklama işleminden dolayı ’’uzun süreli bellek’’ olarak da isimlendirilirler.



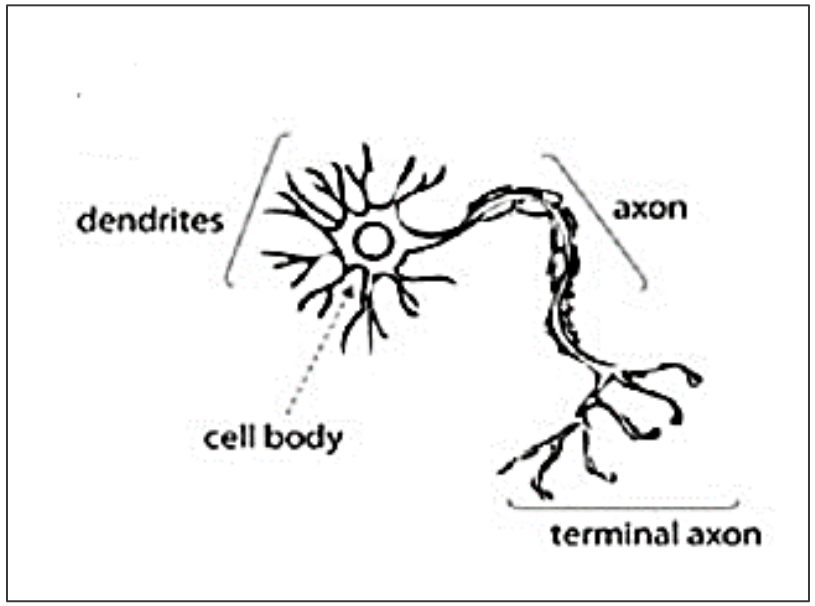
Biyolojik sinir hücresi

- Nöronun bir sinyali alıp, diğer bir nörona iletme süresi yaklaşık 1 milisaniyedir.
- Sinapslar yolu ile gelen sinyal belirli bir eşik seviyesini geçerse iletilir.



Biyolojik sinir hücresi

- Beyindeki mesajların iletim hızı 580 km/saat'e kadar çıkabilmektedir.
- Bilgisayarlarda ise nanosaniyeler mertebesinde çalışmaktadır.
- Beynin karmaşık algılama yapısının hızı sinirlerin hızına bağlı değildir, çok fazla sayıdaki sinir hücresinin paralel çalışmasından kaynaklanmaktadır.



YSA - giriş

- YSA kavramı beynin çalışma ilkelerinin sayısal bilgisayarlar üzerinde taklit edilmesi fikri ile ortaya çıkmıştır.
- İlk çalışmalar beyni oluşturan biyolojik hücrelerin, yada nöronların matematiksel olarak modellenmesi üzerine yoğunlaşmıştır.
- YSA, eğitim ile elde edilen bilgiyi saklamak ve kullanmak için uygun hale getirmeye yarayan, basit işleme elemanlarından oluşur.
- Bilgi, YSA tarafından bir öğrenme işlemi sonucunda elde edilir ve sinaptik ağırlıklar olarak bilinen sinirler arası bağlantılar kullanılarak saklanır.
- YSA'nın hesap gücü; paralel yapısından, öğrenme ve genelleme kabiliyetinden ileri gelir.

YSA - giriş

- YSA
  - Öğrenme
  - İlişkilendirme
  - Sınıflandırma
  - Genelleme
  - Özellik belirleme
  - Optimizasyon
  - vb. konularda başarılı bir şekilde kullanılmaktadır.

YSA - giriş

- Çevre şartları değiştikçe yeni şartları da içeren öğrenme ile YSA yapısı ağırlıklarını güncelleyerek kendini yeni duruma uyarlayabilir.
- Farklı alanlarda aynı YSA yapısı ve teorisi kullanılabilir.
- Kullanım alanları:
    - Proses modelleme ve kontrol(Process Modeling and Control)
    - Makine hatalarının tespiti (Machine Diagnostics)
    - Hedef Tanıma (Target Recognition)
    - Tıbbi Teşhis (Medical Diagnosis)
    - Ses Tanıma (Voice Recognition)
    - Finansal tahmin (Financial Forecasting)
    - Kalite Kontrol (Quality Control)
    - Zeki Arama (Intelligent Searching)

YSA– kullanım alanları

- İşaret işleme: İşaret işleme, elektronik ve bilgisayar mühendisliğinin önemli konularından biridir. Ses ,görüntü, vb. gibi fiziksel büyüklükler bilgisayar ortamına sayısal işaret olarak alınır ve değerlendirilir.
  - YSA işaretlerin alınmasında ve sayısallaştırılmasında hata giderici olarak, veya bulanık dataları düzenleyerek daha kaliteli işaretler elde edilmesini sağlayabilir.
- Kontrol: Kontrol, mühendisliğin en önemli çalışma alanlarından biridir. YSA'ların gelişmesi ile birlikte kontrol alanında yeni ufuklar açılmıştır.
  - Araç pozisyonlarının sezilmesi ve yön belirleme, gemi kontrolü ve park edilmesi, üretim ortamındaki prosesin kontrolü ve otomasyonu, vs.

YSA – kullanım alanları

- Robotik: – Kontrol, robot görmesi, robotlar arası etkileşimli çalışma, vb.
- Örüntü tanıma: el yazısı tanıma, parmak izi tanıma, vb.
- Konuşma üretimi: Yazıların okunması ve telaffuzla ilgili vurguların gerçekleştirilmesi
- Ses tanıma : sayısal ortama aktarılan sesin kime ait olduğunun bulunması

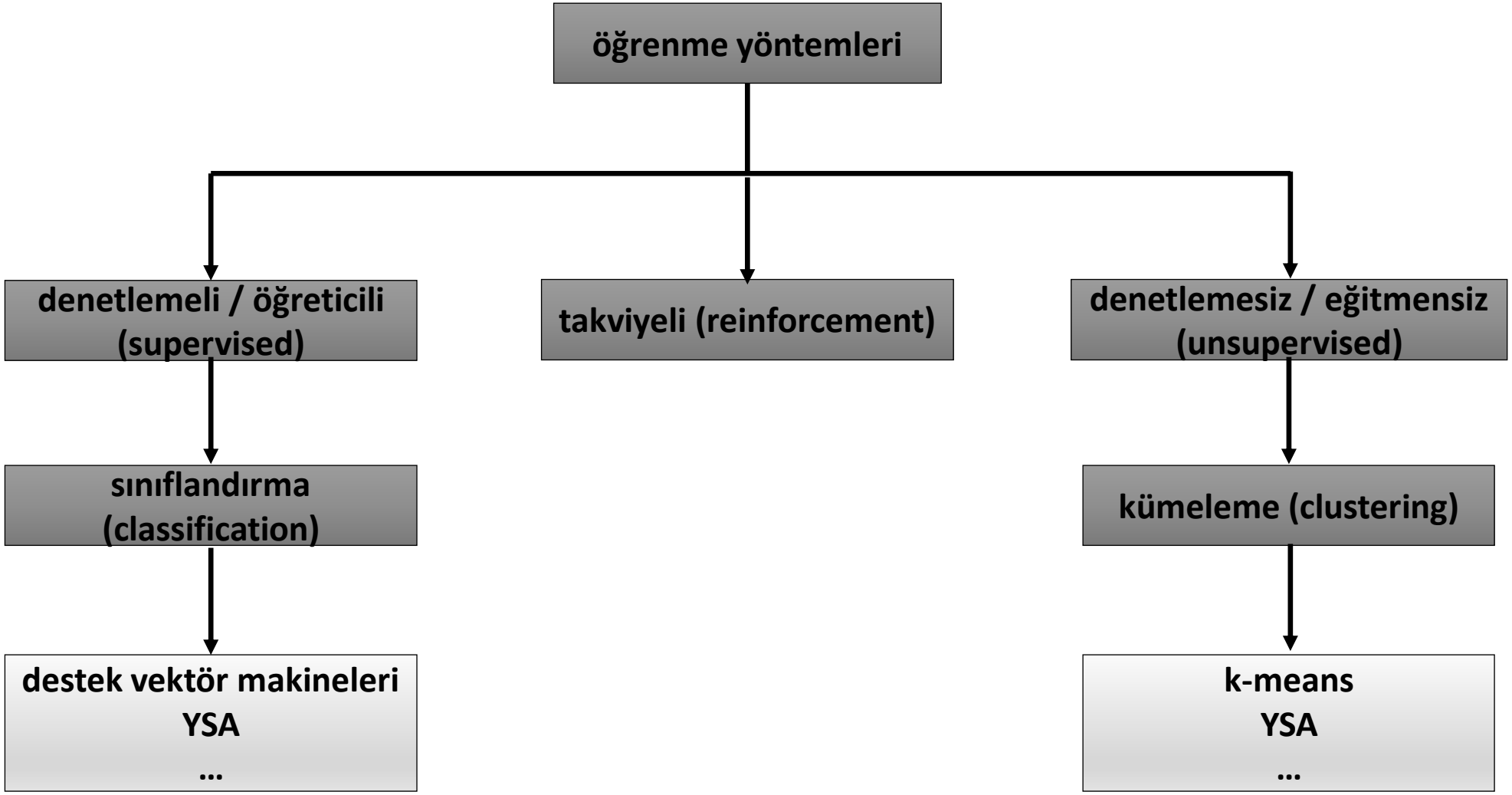
YSA – kullanım alanları

- Bilgisayar görmesi: İnsan yüzü tanıma, kenar belirleme, görsel arama, vb.
- Finansal uygulamalar: Zamana bağlı olarak analizler, borsa tahminleri, vb.
- Veri sıkıştırma: Ses sinyallerinin ve resim datalarının sıkıştırılması
- Oyunlar: oyun programlarında strateji belirleme ve hamlelerin belirlenmesi
- …

YSA'da öğrenme

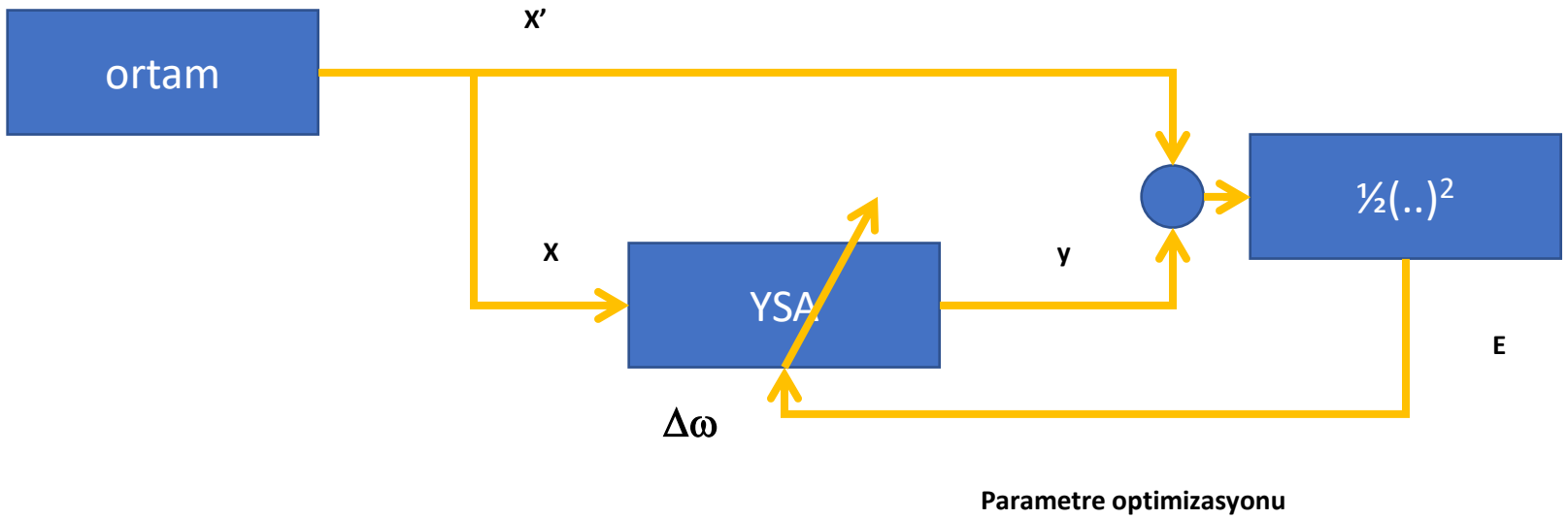
- YSA'da öğrenme işlemi, belirli bir problemin optimum çözümünü sağlayan ağırlıklarını bulmaya çalışmaktır.
- Bu ağırlıklar adım adım güncellenerek problem için sonuçlandırılır.
  - Gradyan bazlı öğrenme algoritmaları:
    - Bütün gradyan bazlı eğitim algoritmaları global minimumu garanti etmez.
    - Fakat bazı metotlar diğerlerine göre global minimuma yakın lokal minimumlara ulaşır ve bazıları değerine göre daha hızlı yakınsar.
    - Gradyan bazlı algoritmalarda sıradaki nokta deterministik hesap ile bulunur. Deterministik: Bir problemde sonuca kaç adımda ulaşılacağı ve belirli bir iterasyon sonrası sonucu önceden hesaplanabiliyorsa bu problem deterministiktir.

YSA'da öğrenme



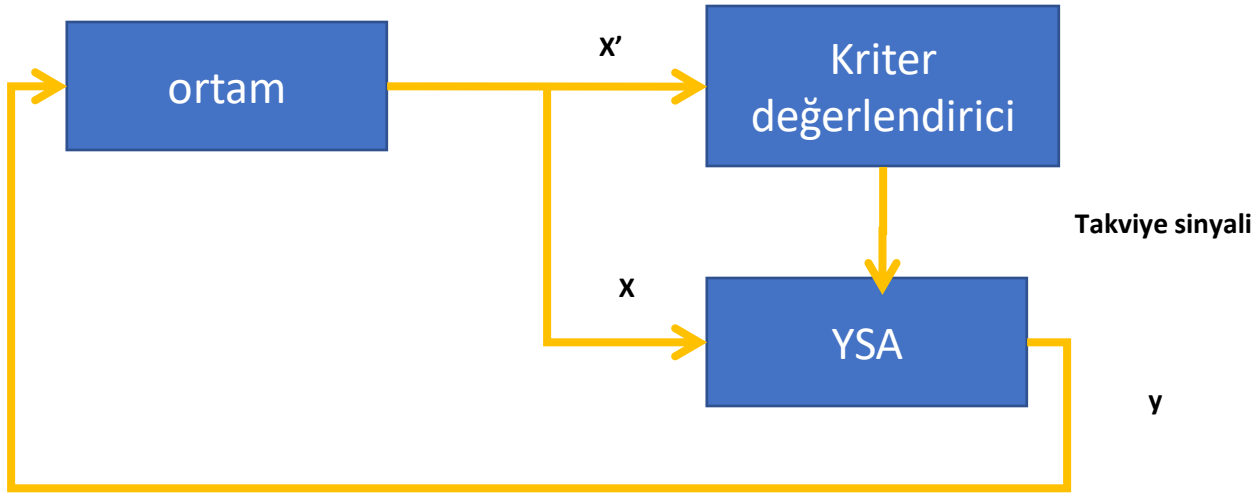
YSA'da öğrenme

- Destekli(supervised) öğrenme/ Öğreticili öğrenme
  - Öğrenilmesi istenen problem/sistem ile ilgili örnekler girdi/çıktı seti olarak verilir.
  - Burada YSA'nın görevi; girdileri, eğitim setinde verilen çıktılara haritalamadır / eşleştirmedir.
  - Back propoagation(geriye yayılım), vb. türev bazlı öğrenmeler bu gruptandır.



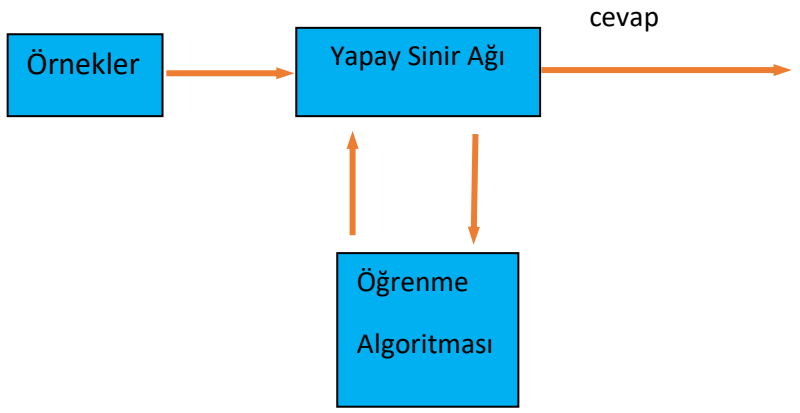
YSA'da öğrenme

- Takviyeli Öğrenme
  - Öğreticili öğretmeden farklı olarak, üretmesi gereken sonuca göre sadece doğru veya yanlış bilgisinin ağa bildirilmesidir. Bu ise ağa bir takviye sinyalinin gönderilmesi ile gerçekleşir.
  - Girdiler karşısında istenilen değerler yerine istenilen değerlere yaklaşıp yaklaşmadığını belirtilir.
  - Sistem öğreticiden gelen bu sinyallere göre öğrenme sürecini gerçekleştirir.

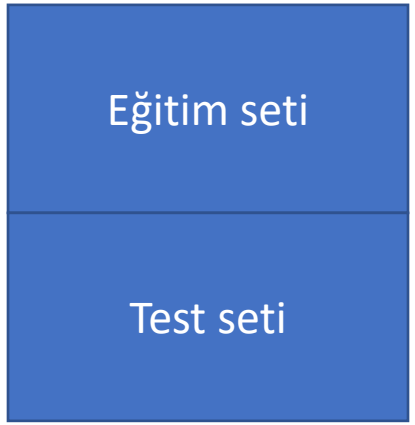


YSA’da öğrenme

- Öğretmensiz Öğrenme
  - Bu durumda hiçbir öğretmene ihtiyaç yoktur. Dolayısıyla, bu tür öğrenme şekline ‘kendi kendine organize etme’ de denilmektedir. Ağ, kendisine gösterilen örnekleri alır ve belli kriterlere göre gruplar. Bu kriter önceden bilinmeyebilir. Ağ kendi öğrenme kriterini kendisi oluşturur.
  - Sisteme(YSA) sadece girdi değerleri gösterilir.
  - Ağın serbest parametreleri girdi değerlerine göre optimize edilir.
  - Ağ parametrelerinin güncellenmesi verilerin (giriş) istatistiksel düzenine göre gerçekleştirilir.



YSA’da öğrenme - genelleme



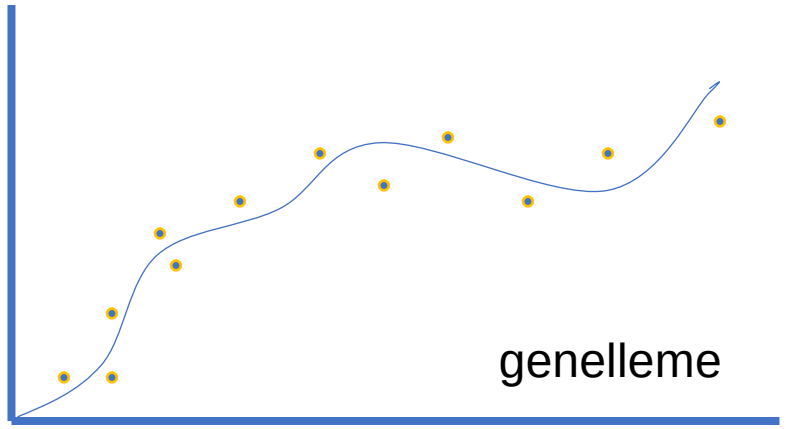
Eğitimde kullanılır

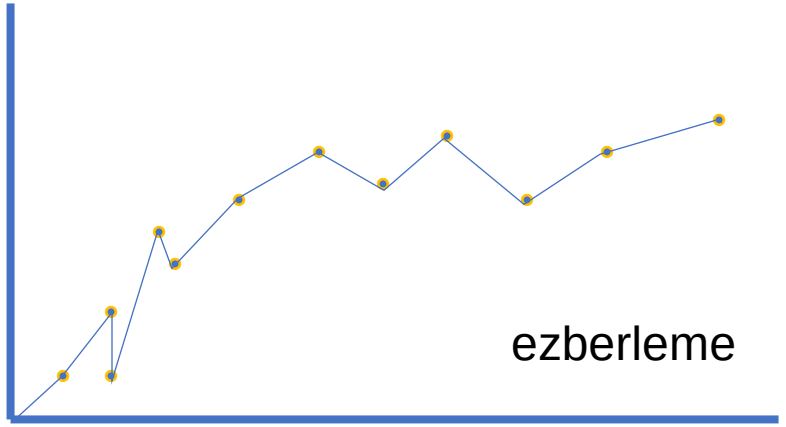
Ağın öğrenip öğrenmediğini test etmede kullanılır

- YSA tarafından eğitim esnasında kullanılan sayısal değerlerden giriş çıkış eşleştirmesini belirleyen temel ilişkinin çıkarılması ve böylelikle eğitim esnasında kullanılmayan girdiler içinde anlamlı yanıtlar çıkarılması kabiliyetidir.
- YSA eğitim işlemi sırasında amaç daha önce ağın görmediği girişler için ağın anlamlı yanıtlar vermesidir.

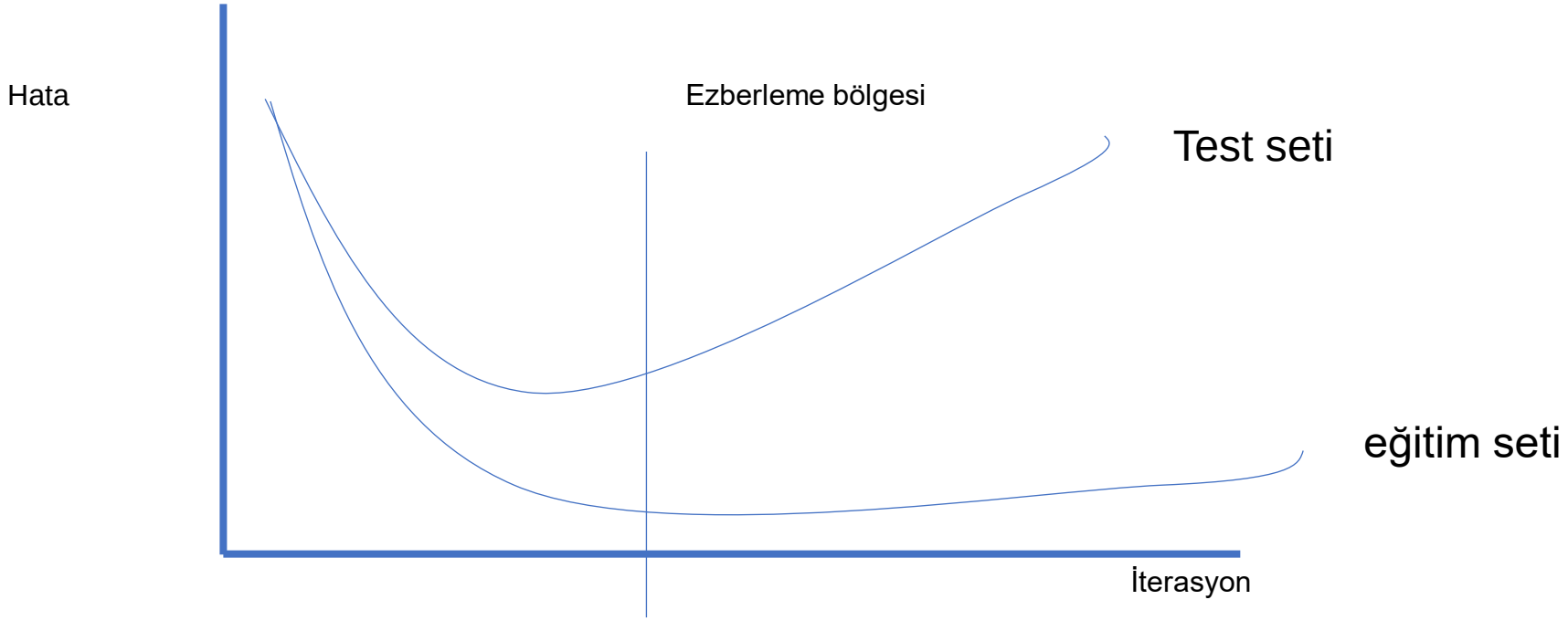
YSA’da öğrenme - ezberleme

- YSA eğitim setine iyi sonuçlar veriyor ve test setine uygun sonuç vermiyorsa YSA eğitim setini ezberlemiştir.
- Aşırı eğitim ve yetersiz eğitim ezberlemeye sebep olabilir.





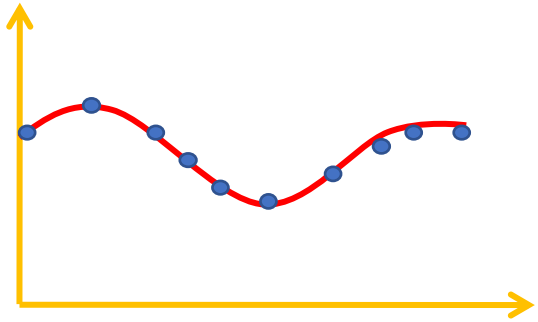
YSA’da öğrenme - ezberleme



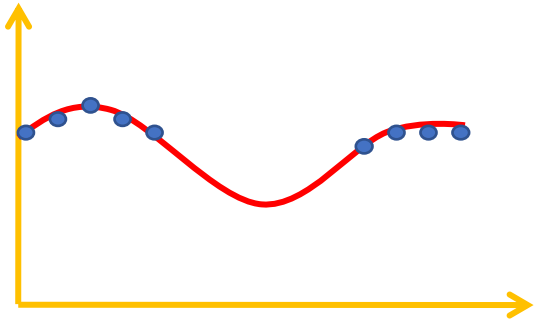
YSA’da öğrenme – eğitimde örnek seçimi

- Seçilen örnekler problem uzayını iyi temsil etmelidir.
- Eğitilmiş YSA, eğitim setinde ve test setinde başarılı ise öğrenme gerçekleşmiştir.
  - Fakat buna rağmen kullanım ortamında başarısız ise eğitim ve test setlerinde kullanılan örnekler problem uzayını iyi temsil etmiyor demektir.
  - Bu durumda eğitim ve test örnekleri geliştirilmeli.
- YSA’da giriş ve çıkışların sayısal gösterilmesi şarttır.
- Giriş değerleri özellik çıkarma (feature extraction) gibi ön işleme yapılarak azaltılabilir. Bu durumda performans düşebilir fakat işlem yükü azalır.

YSA’da öğrenme – eğitimde örnek seçimi

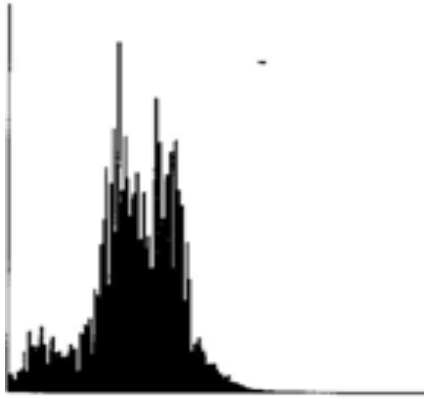


Uygun örnek seçimi

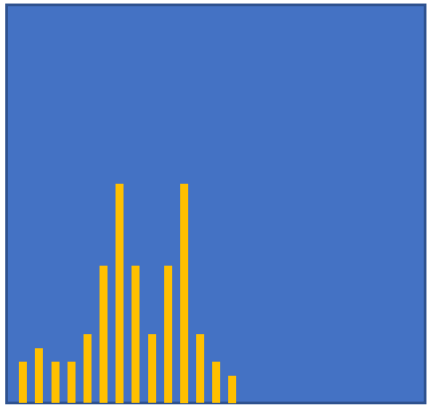


Uygun olmayan örnek seçimi

256x256 giriş



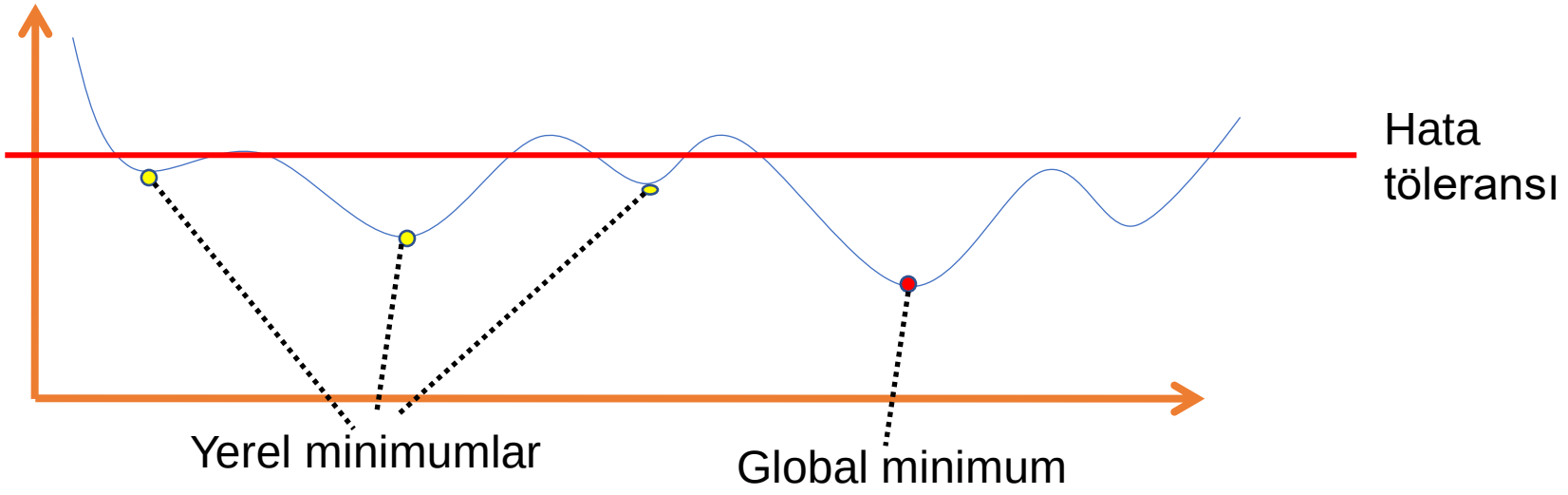
256 giriş



14 giriş

YSA'da öğrenme - momentum ve öğrenme katsayısının önemi

- YSA kabul edilebilir çözümler üretir. En iyi global çözümü garanti etmeyebilir.
  - Momentum katsayısı ağın yerel minimuma takılmasını önleme amaçlı kullanılır.
  - Öğrenme katsayısı ağırlıkların güncellenmesinde değişim miktarını belirler.



YSA’da öğrenme

- Öğrenmenin yeterli olmamasının sebepleri:
  - Eğitim algoritmaları yerel minimumlara takılabilir.
  - Eğitim seti problem uzayını iyi bir şekilde temsil etmeyebilir.
  - Uygun parametreler ve başlangıç ağırlıkları seçilmemiştir.
  - YSA topolojisi problem için yeterli değildir.

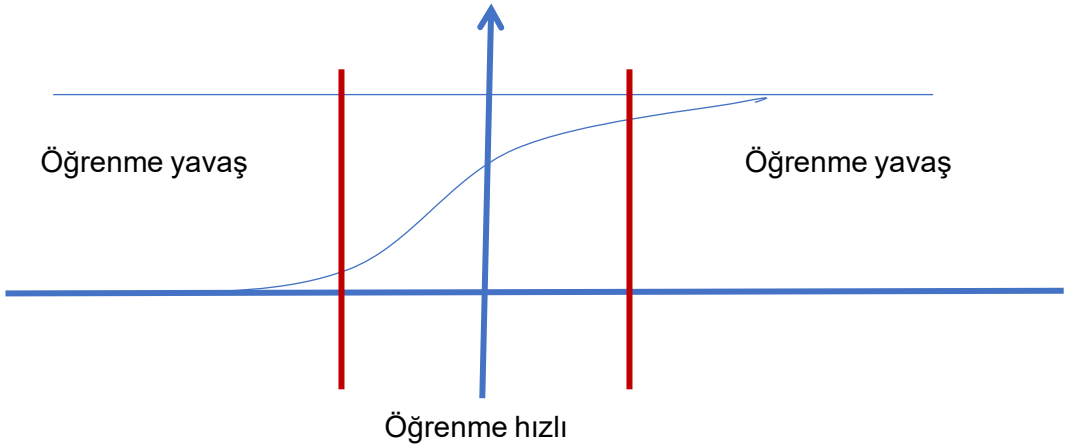
YSA'da öğrenme

- Çevrimiçi(online) / eş zamanlı öğrenme
  - Parametre güncelleme işleminin, normal çalışma esnasında anlık gözlemlerden alınan bilgiler ile yayılması
- Çevrimdışı (offline) öğrenme
  - Daha önceden belirlenen giriş/çıkış eşleştirmesini gerçekleştirme
- Ölçeklendirme / Normalizasyon
  - Problem uzayının girişleri farklı ölçekler kullanan ortamlardan oluşabilir.
  - Ölçeklendirildikten sonra bütün girdiler belirli bir aralığa dahil olmaktadır (genellikle 0-1 aralığı).

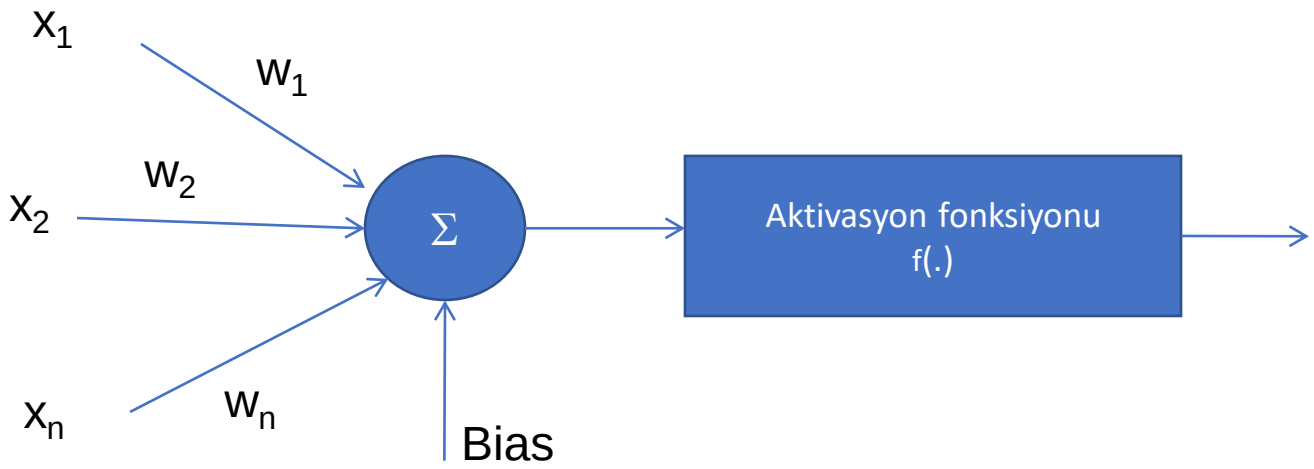
$$\widehat{x} = \frac{x - x_{\min}}{x_{\max} - x_{\min}}$$

YSA’da öğrenme

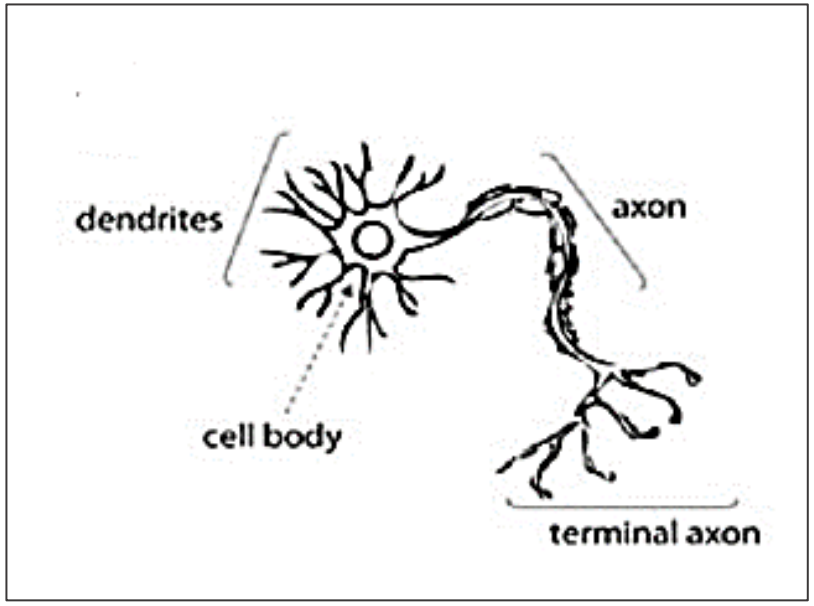
- Ölçeklendirme / Normalizasyon
  - Normalizasyonun diğer bir avantajı da normalizasyon sayesinde girişlerle, girişleri sonraki katmana bağlayan ağırlıklar vektörlerinin başlangıçta küçük değerli ağırlıklarda olduğu için aktivasyon fonksiyonunun aktif bölgesine çekilmesidir. Ör: sigmoid



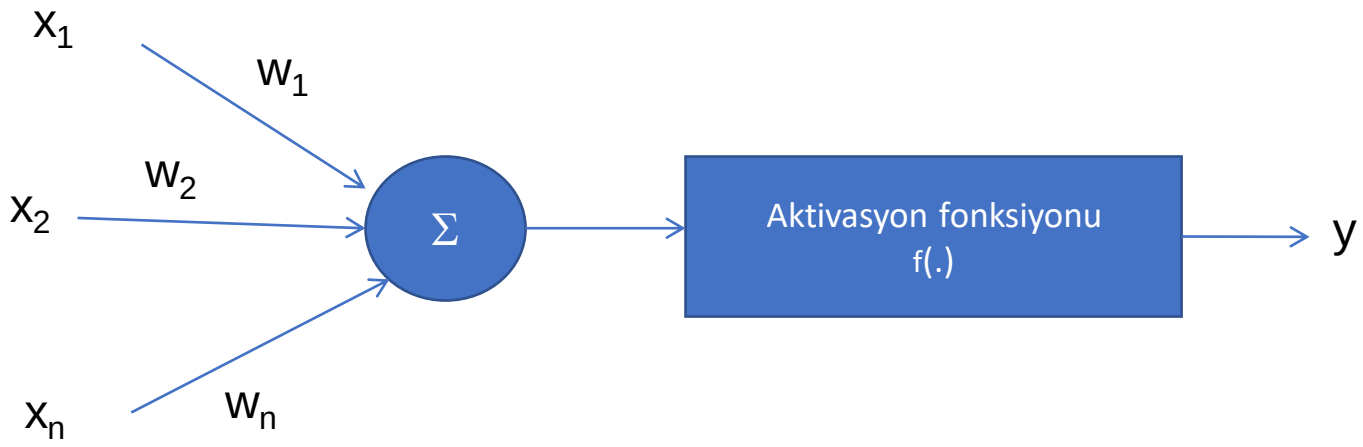
Yapay sinir hücresi



- Yapay sinir; biyolojik sinirin girdi, işlem ve çıktı karakteristiğini taklit etmek için tasarlanmıştır.
- Burada her bir girdi kendi ağırlığı ile çarpılmakta ve bu çarpımların hepsi toplanmaktadır.
- Bu toplam sinaptik kuvvete benzetilebilir ve nöronun aktivitesini belirlemek için kullanılır



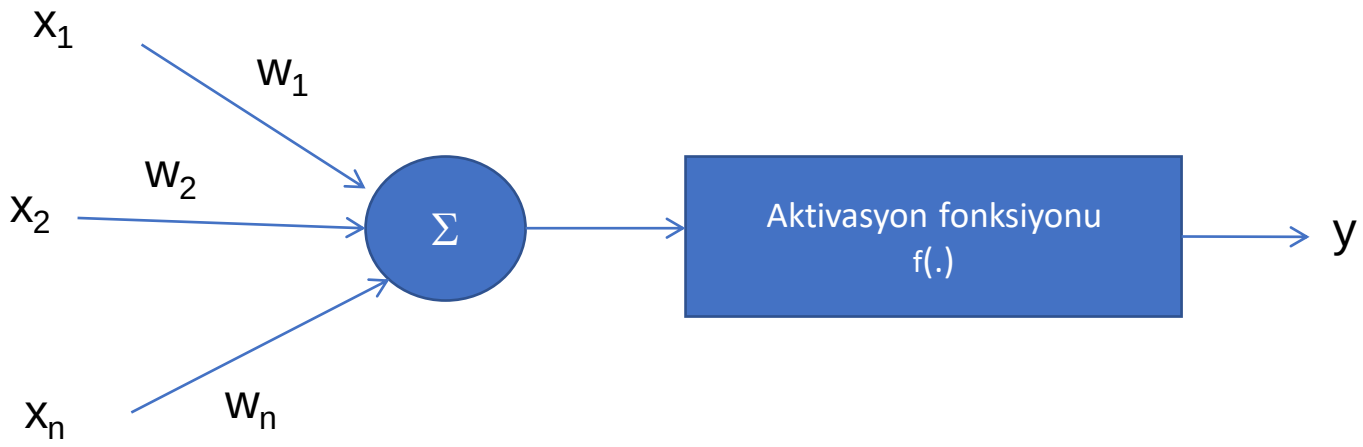
Yapay sinir hücresi



$$\boldsymbol{net} = \boldsymbol{x_1 w_1} + \boldsymbol{x_2 w_2} + \ldots + \boldsymbol{x_n w_n}$$

$$= \sum_{\boldsymbol{i=1}}^{\boldsymbol{n}} \boldsymbol{x_i w_i}$$

Yapay sinir hücresi



$$\boldsymbol{y = f(net)}$$

$$= \begin{cases} \mathbf{1}, & \boldsymbol{net \geq t} \\ \mathbf{0}, & \boldsymbol{net < t} \end{cases}$$

(örnek fonksiyon)

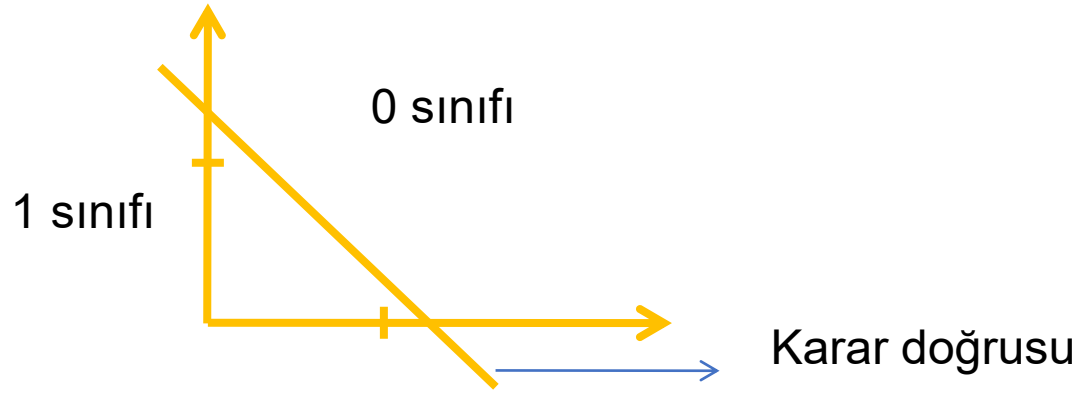
Yapay sinir hücresi

- YSA'da *net* sinyali genellikle aktivasyon fonksiyonu *f* ile işlem görerek nöronun çıkış sinyalini oluşturur.
- YSA'da kullanılacak olan aktivasyon fonksiyonuna problemin yapısına göre karar verilir.
- Daha önceki deneyimlerden yararlanılarak uygun aktivasyon fonksiyonu seçilebilir.
- Örnek fonksiyonlar: $\boldsymbol{f(x) = x}$ .....lineer fonksiyon

$$\boldsymbol{f(x) = \frac{1}{1+e^{-x}}}$$ .....sigmoidal fonksiyon

Perceptron (algılayıcı)

- Tek katmanlı algılayıcılar problem uzayını bir doğru, düzlem veya hiper düzlem ile doğrusal olarak sınıflandırılabilir.
- Doğrusal olmayan sınıflama işlemlerini yapamazlar. Noonlineer sınıflandırma için çok katmanlı algılayıcılar geliştirilmiştir.
- Örnek ayrıştırma problemi: NAND problemi

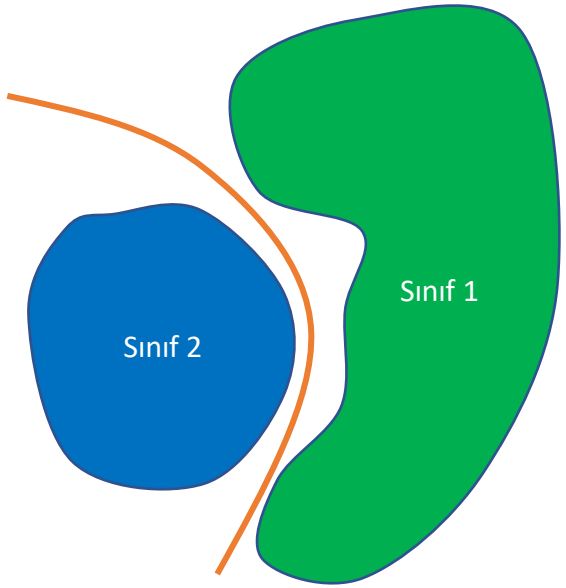


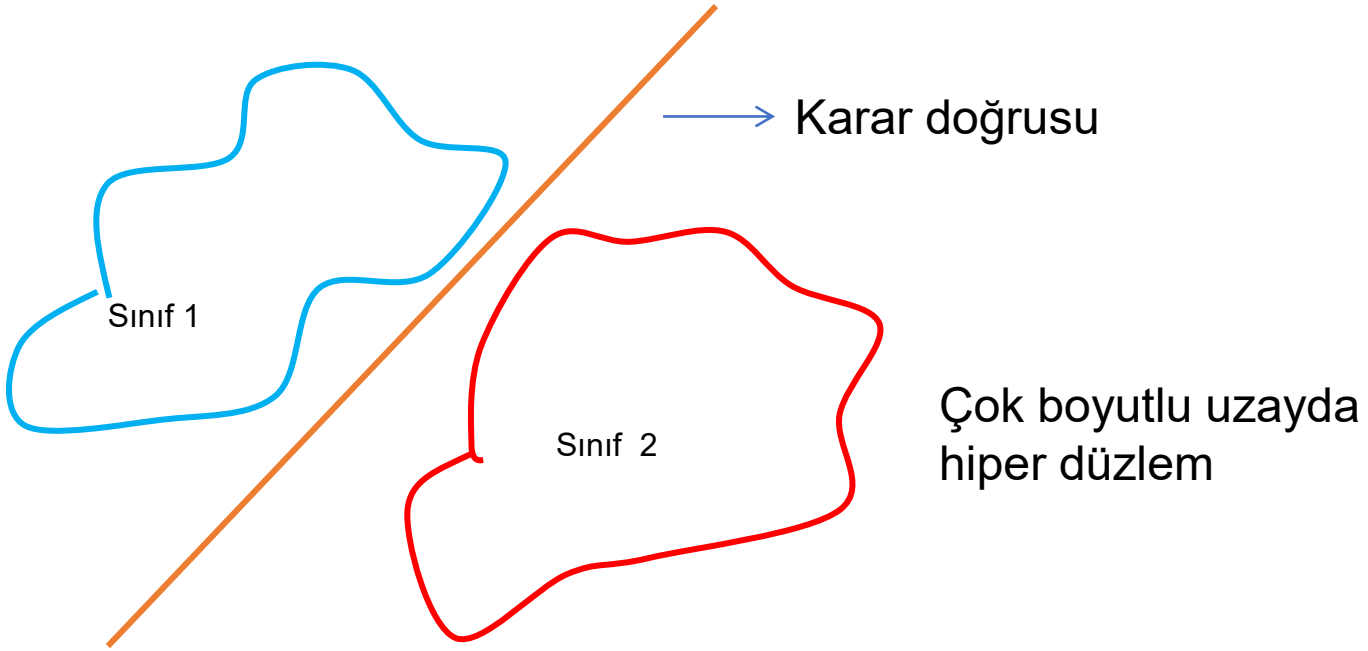
A	B	S
0	0	1
0	1	1
1	0	1
1	1	0

- Doğrusal ayrıştırılabilir. Tek perceptron yeterli

Perceptron (algılayıcı)

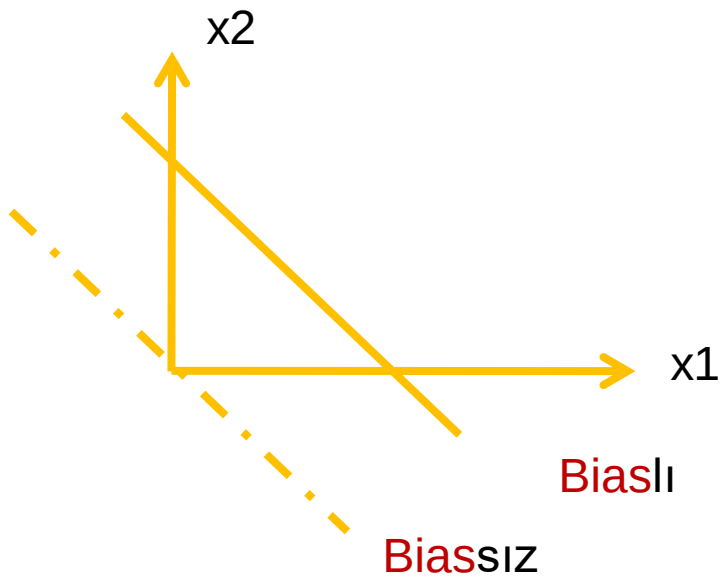
- Perceptronun uygun bir şekilde sınıflandırılabilmesi için, ayırt edilecek iki sınıfın doğrusal olarak ayırt edilebilmesi gereklidir.



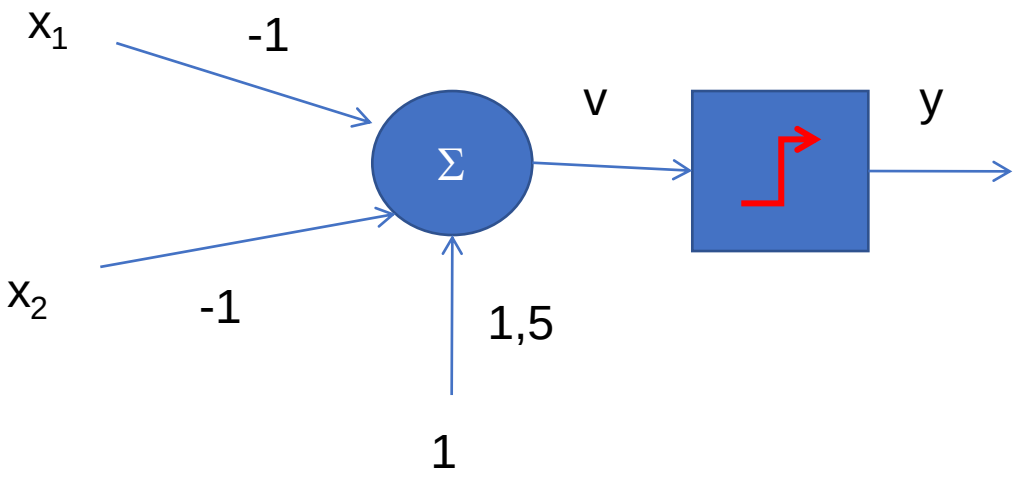


Perceptron (algılayıcı)

- Karar doğrusunda bias doğrunun orijinden kaydırılmasını sağlar.
- Ağırlıklar eğrinin eğimini belirler.



Lineer NAND problemi



$$Y = \begin{cases} 1 \text{ eğer } v>0 \\ 0 \text{ eğer } v<=0 \end{cases}$$

A	B	S
0	0	1
0	1	1
1	0	1
1	1	0

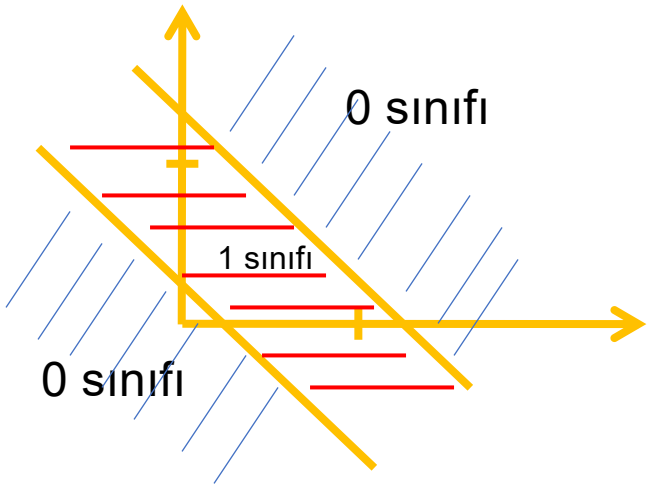
- $v = -x_1 - x_2 + 1,5$

- -1(0)-1(0)+1,5=1,5
  - ise y=1
- -1(0)-1(1)+1,5=0,5
  - ise y=1
- -1(1)-1(0)+1,5=0,5
  - ise y=1
- -1(1)-1(1)+1,5=-0,5
  - ise y=0

Doğrusal olmayan örnek problem: XOR

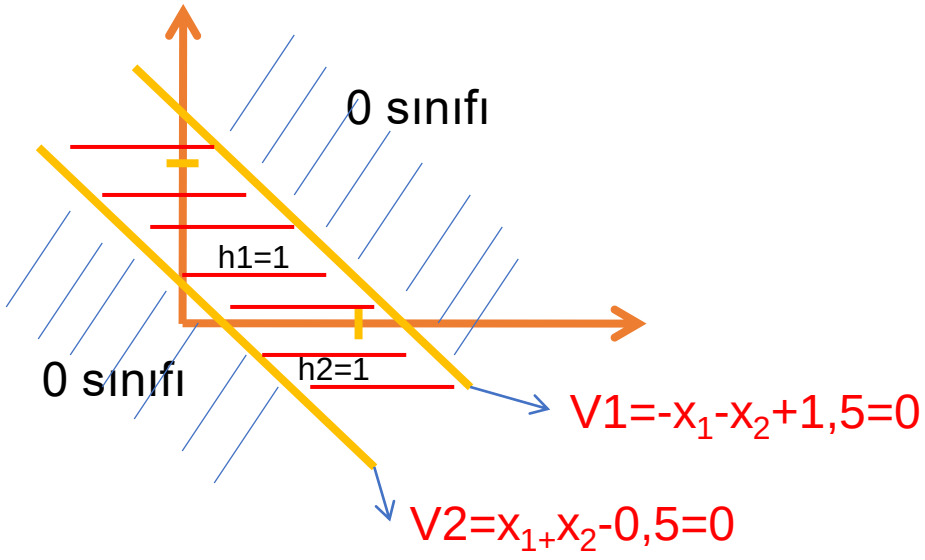
- XOR problemi; doğrusal ayrıştırılamayan problemdir, tek katmanlı perceptronlar çözemez.
- XOR probleminin önemi; tek katmanlı perceptronların bu problemi çözemeyeceği gösterilmesi ile YSA ile ilgili çalışmalar durma noktasına gelmiştir.
- Çok katmanlı perceptron ile bu problemin çözülmesi YSA'ya olan ilgiyi yeniden artırmıştır.
- XOR problemi doğrusal olmayan bir problemi/ilişkiyi ifade ettiği için doğrusal olmayan problemleri temsil etmektedir.

Doğrusal olmayan örnek problem: XOR



A	B	S
0	0	0
0	1	1
1	0	1
1	1	0

Doğrusal olmayan örnek problem: XOR



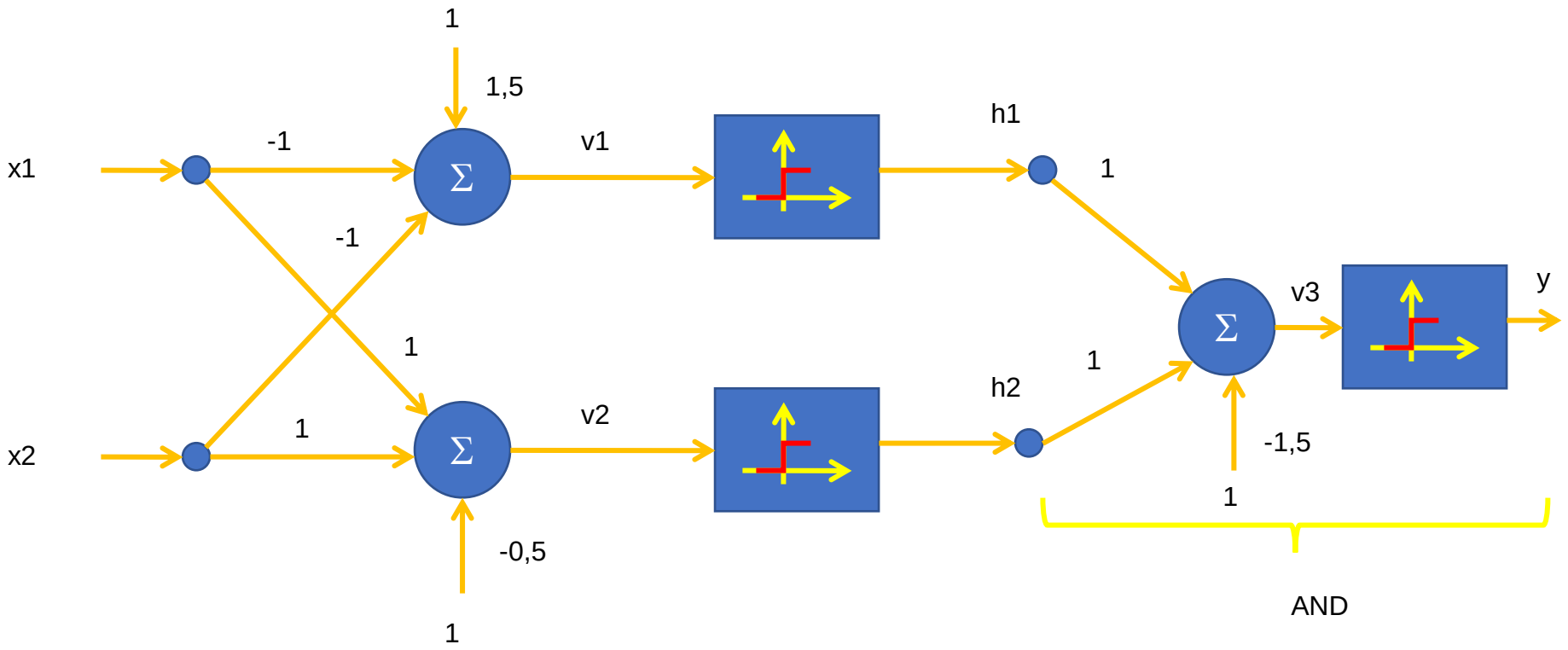
$V1=-x_1-x_2+1,5=0$

$V2=x_{1+}x_2-0,5=0$

$v3=h_1+h_2-1,5=0$

Doğrusal olmayan örnek problem: XOR

A	B	S
0	0	0
0	1	1
1	0	1
1	1	0



x1	x2	v1	v2	h1	h2	v3	y
0	0	1,5	-0,5	1	0	-0,5	0
0	1	0,5	0,5	1	1	0,5	1
1	0	0,5	0,5	1	1	0,5	1
1	1	-0,5	1,5	0	1	-0,5	0

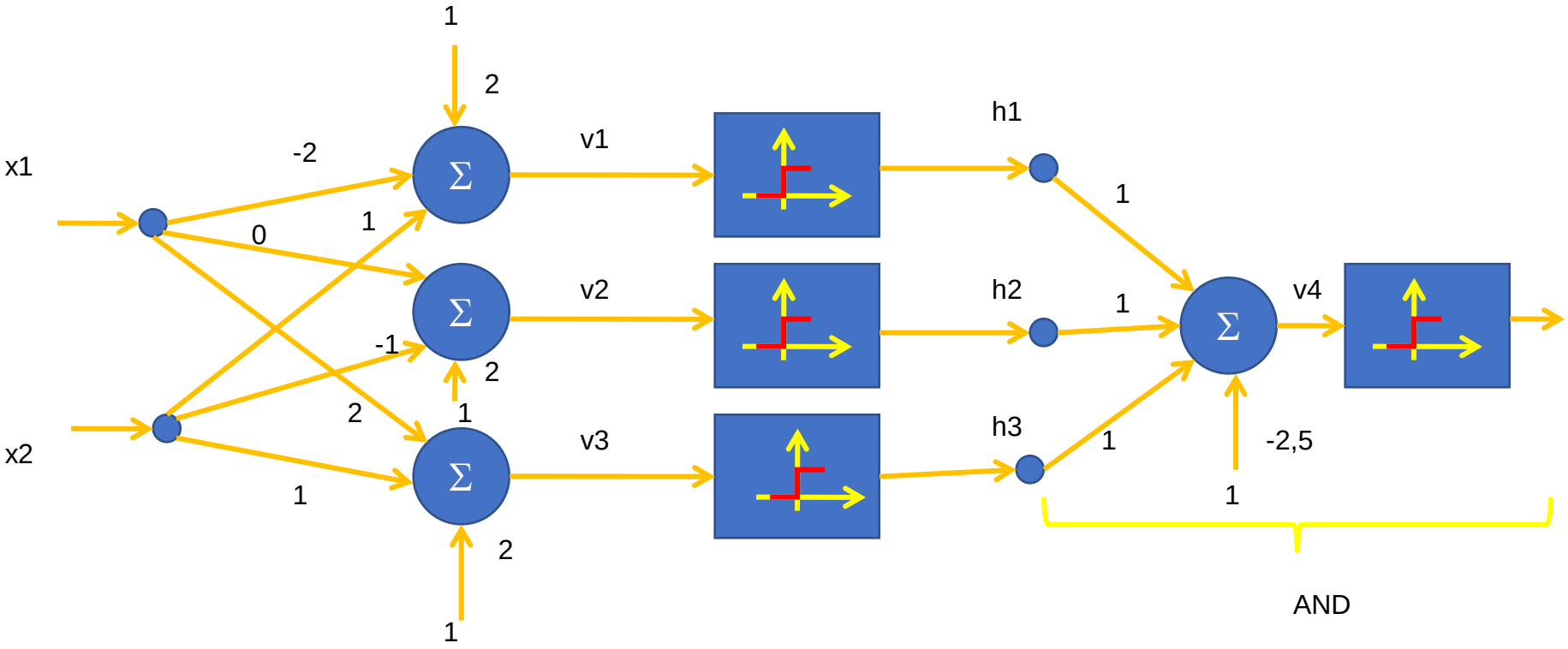
Çok Katmanlı İleri Beslemeli Yapay Sinir Ağları

- Çok katmanlı ağlar denetlemeli öğrenme kullanılarak yapılan ağırlık güncellemesi ile doğrusal olmayan bir giriş çıkış ilişkisine (y=f(x)) yakınsama yapar.
- Çok katmanlı ileri beslemeli yapay sinir ağlarında genellikle sigmoid türü aktivasyon fonksiyonları kullanılır.
- Diğer aktivasyon fonksiyonları da yer yer bir alternatif olarak kullanılmaktadır.
- Dikkat edilmesi gereken durumlardan birisi türev bazlı öğrenme kurallarının uygulanabilmesi için aktivasyon fonksiyonunun türevinin alınabilir olması gerekmektedir.

Çok Katmanlı İleri Beslemeli Yapay Sinir Ağları

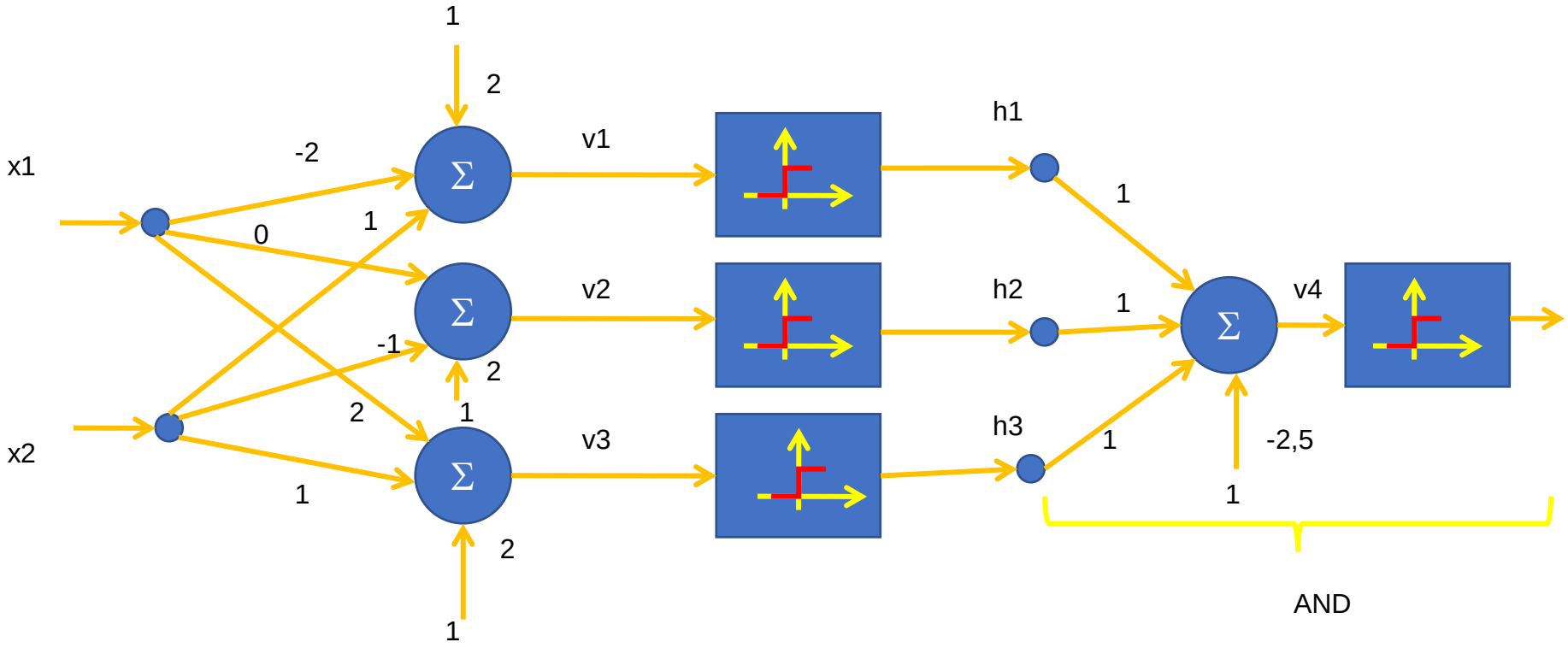
- Sigmoid türü fonksiyonlar, ileri beslemeli yapay sinir ağlarında kullanılan türevi alınabilir tipik doğrusal olmayan fonksiyonlardır.
- Sigmoid tipi aktivasyon fonksiyonlarının özellikleri;
  - Sigmoid fonksiyonlarının türevleri kolaylıkla basit aritmetik işlemler kullanılarak çıkış sinyallerinden elde edilebilirler.
  - Sigmoid tipi aktivasyon fonksiyonlarının türevleri hiç bir zaman negatif olmaz.
  - Türevler öğrenme kurallarında kolaylıkla kullanılmaktadır.

Çok Katmanlı İleri Beslemeli Yapay Sinir Ağları



x1	x2	v1	v2	v3	h1	h2	h3	v4	y
0	0								
0	1								
1	0								
1	1								

Çok Katmanlı İleri Beslemeli Yapay Sinir Ağları

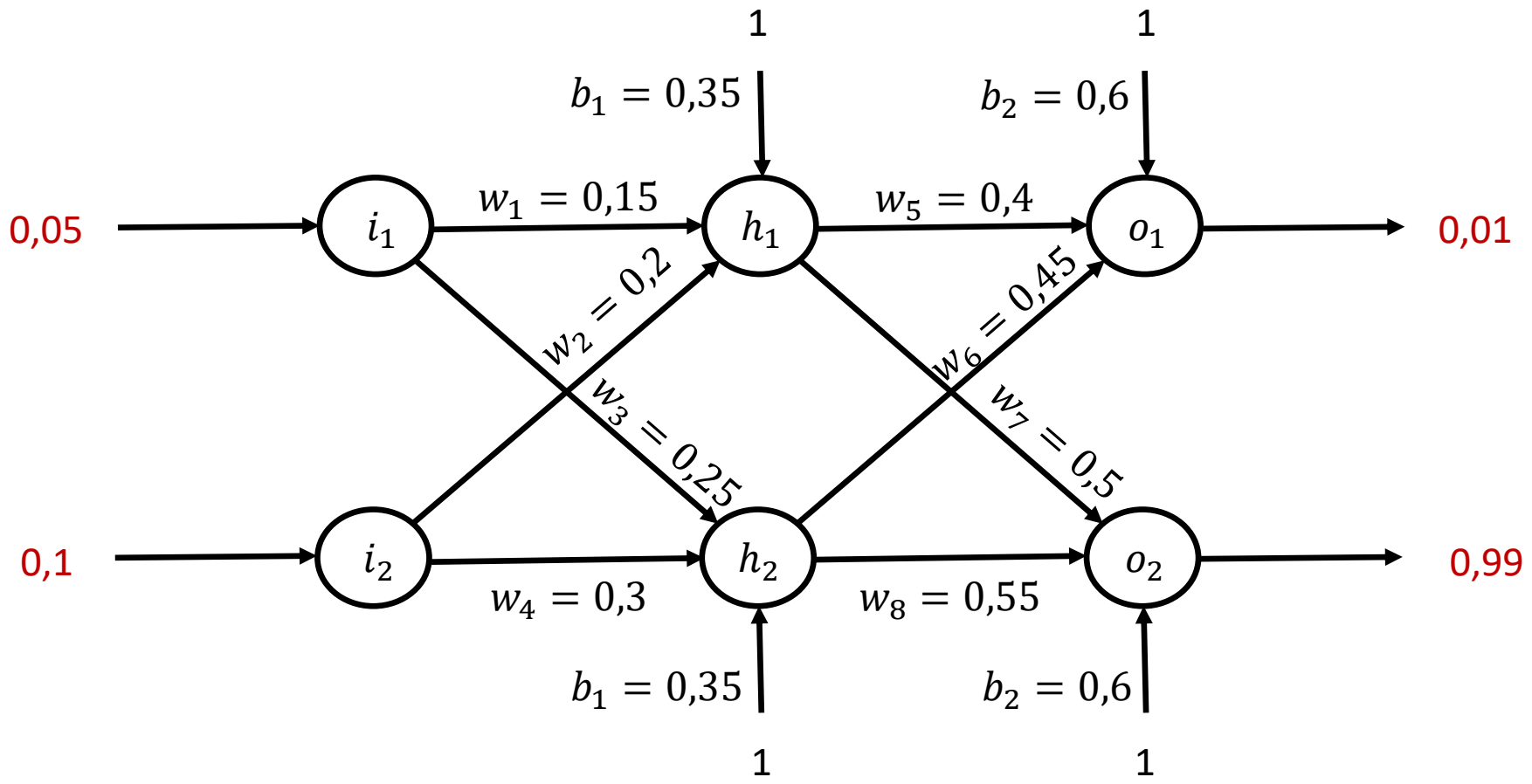


x1	x2	v1	v2	v3	h1	h2	h3	v4	y
0	0	2	2	2	1	1	1	0,5	1
0	1	3	1	3	1	1	1	0,5	1
1	0	0	2	4	0	1	1	-0,5	0
1	1	1	1	5	1	1	1	0,5	1

Geri Yayılım Algoritması

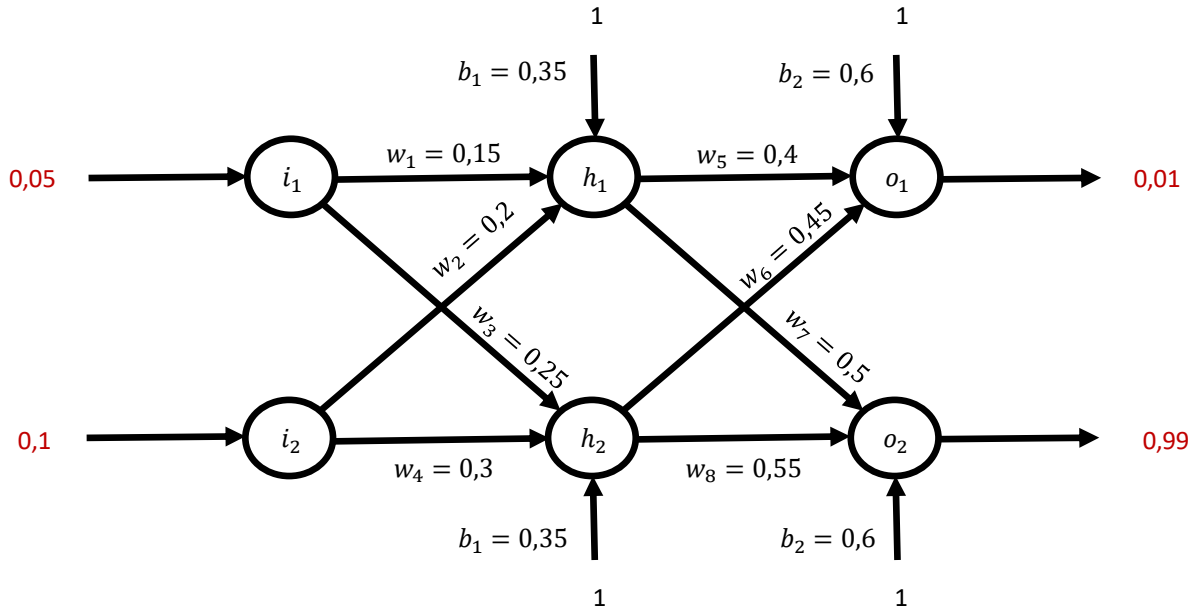
- Geri yayılım algoritması, ileri beslemeli yapay sinir ağları için çok popüler ve önemli bir algoritmadır.
- Bu popülariteden dolayı ileri beslemeli yapay sinir ağları sıklıkla geriye yayılım sinir ağları (back propagation neural networks) olarak da adlandırılırlar.
- Geriye yayılım algoritması, denetlemeli öğrenme algoritmalarındandır.
- İstenen çıkışlar ile gerçek çıkışlar arasındaki hata gizli katmanlara doğru geriye yayılır.

Geri Yayılım Algoritması



Geri Yayılım Algoritması

(ileri yayılım hesaplamaları)



- $net_{h1} = w_1 * i_1 + w_2 * i_2 + b_1 * 1$

  $= 0{,}15 * 0{,}05 + 0{,}2 * 0{,}1 + 0{,}35 * 1$

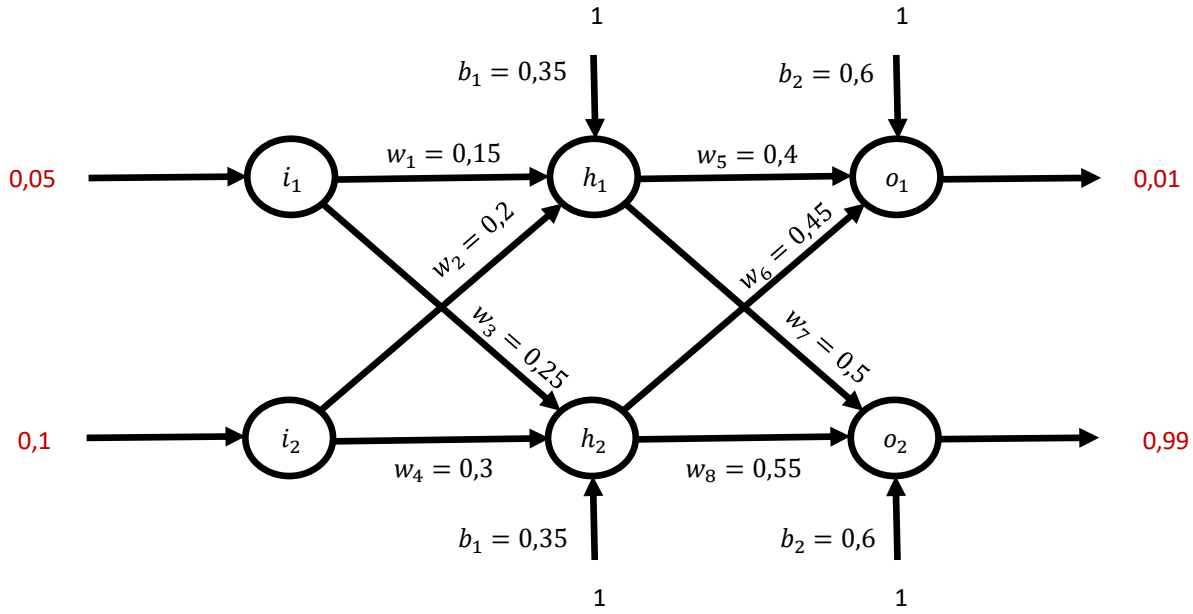
  $= 0{,}3775$

- $out_{h1} = \frac{1}{1+e^{-net_{h1}}}$

  $= \frac{1}{1+e^{-0{,}3775}} = 0{,}5932$

Geri Yayılım Algoritması

(ileri yayılım hesaplamaları)



- $net_{h2} = w_3 * i_1 + w_4 * i_2 + b_1 * 1$

  $= 0{,}25 * 0{,}05 + 0{,}3 * 0{,}1 + 0{,}35 * 1$

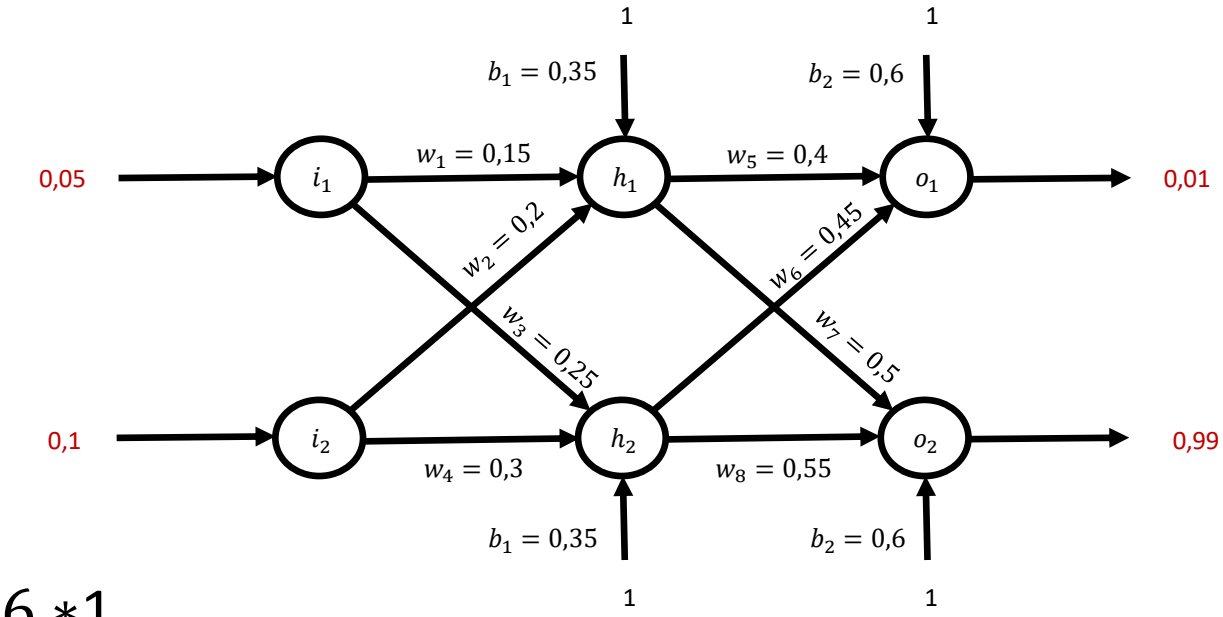
  $= 0{,}3925$

- $out_{h2} = \frac{1}{1+e^{-net_{h2}}}$

  $= \frac{1}{1+e^{-0{,}3925}} = 0{,}5968$

Geri Yayılım Algoritması

(ileri yayılım hesaplamaları)



- $net_{o1} = w_5 * out_{h1} + w_6 * out_{h2} + b_2 * 1$

  $= 0{,}4 * 0{,}5932 + 0{,}45 * 0{,}5968 + 0{,}6 * 1$

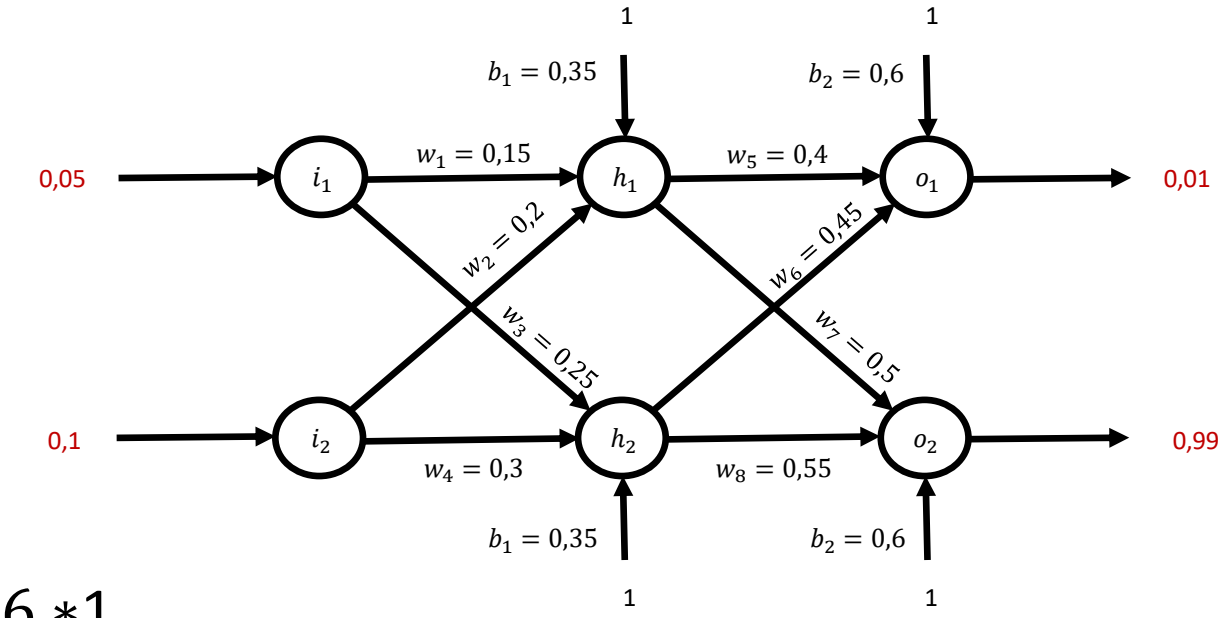
  $= 1{,}1059$

- $out_{o1} = \frac{1}{1+e^{-net_{o1}}}$

  $= \frac{1}{1+e^{-1{,}1059}} = 0{,}7513$

Geri Yayılım Algoritması

(ileri yayılım hesaplamaları)



- $net_{o2} = w_7 * out_{h1} + w_8 * out_{h2} + b_2 * 1$

  $= 0{,}5 * 0{,}5932 + 0{,}55 * 0{,}5968 + 0{,}6 * 1$

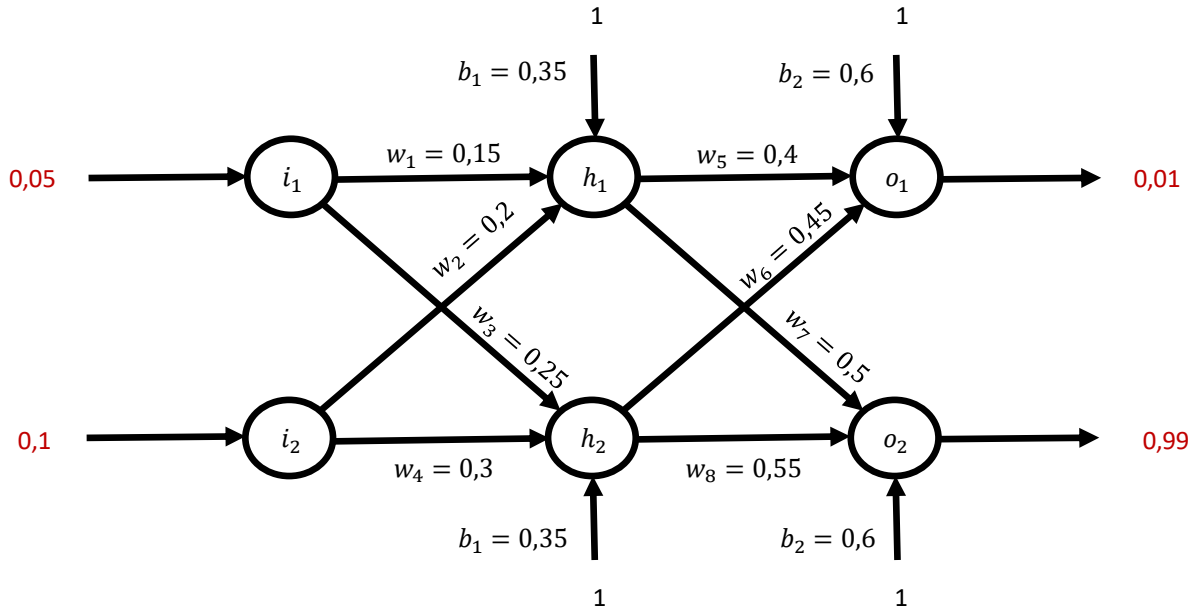
  $= 1{,}2248$

- $out_{o2} = \frac{1}{1+e^{-net_{o2}}}$

  $= \frac{1}{1+e^{-1{,}2248}} = 0{,}7729$

Geri Yayılım Algoritması

(toplam hata hesaplamaları)



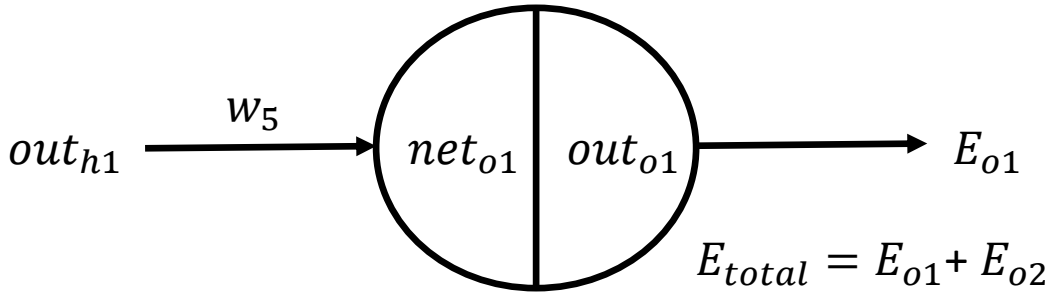
- $E_{total} = \Sigma \frac{1}{2}(target - output)^2$

- $E_{o1} = \frac{1}{2}(target_{o1} - out_{o1})^2 = \frac{1}{2}(0{,}01 - 0{,}7513)^2 = 0{,}2748$
- $E_{o2} = \frac{1}{2}(target_{o2} - out_{o2})^2 = \frac{1}{2}(0{,}99 - 0{,}7729)^2 = 0{,}0235$

- $E_{total} = E_{o1} + E_{o2} = 0{,}2748 + 0{,}0235 = 0{,}2983$

Geri Yayılım Algoritması

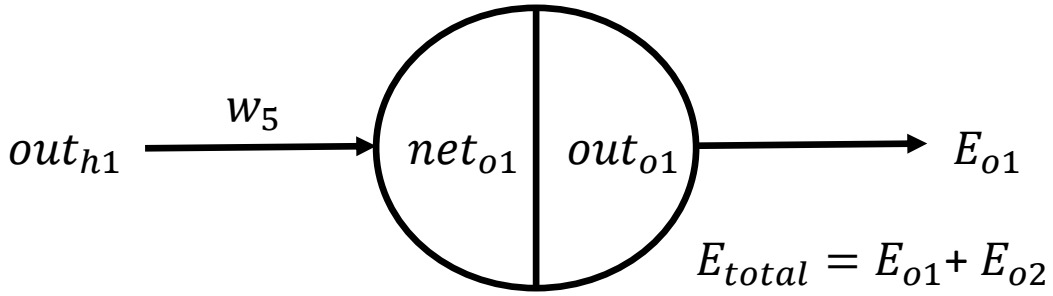
- w_5 ağırlığındaki bir değişiklik toplam hatayı ne kadar etkiler?



- E_{total} 'in w_5 'e göre kısmi türevi: $\frac{\partial E_{total}}{\partial w_5}$

- Zincir kuralı (chain rule): $\frac{\partial E_{total}}{\partial w_5} = \frac{\partial net_{o1}}{\partial w_5} \frac{\partial out_{o1}}{\partial net_{o1}} \frac{\partial E_{total}}{\partial out_{o1}}$

Geri Yayılım Algoritması

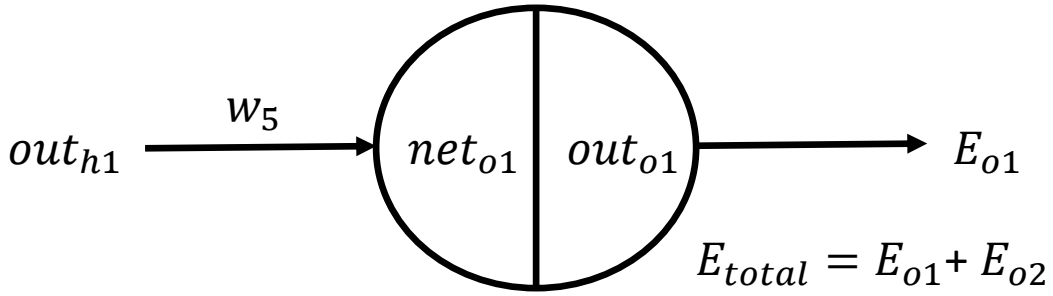


- $\frac{\partial E_{total}}{\partial out_{o1}} = (\frac{1}{2}(target_{o1} - out_{o1})^2 + \frac{1}{2}(target_{o2} - out_{o2})^2)'$

  $= 2\frac{1}{2}(target_{o1} - out_{o1})(-1) + 0 = out_{o1} - target_{o1}$

  $= 0{,}7513 - 0{,}01 = 0{,}7413$

Geri Yayılım Algoritması



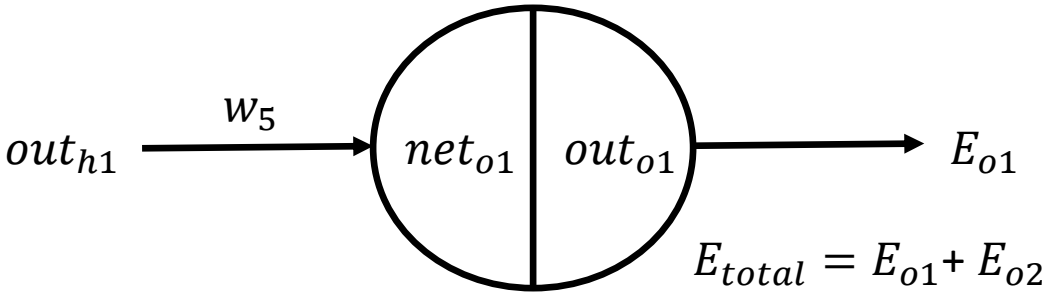
- $\frac{\partial out_{o1}}{\partial net_{o1}} = ?$

- $out_{o1} = \frac{1}{1+e^{-net_{o1}}} \rightarrow (out_{o1})' = \frac{e^{-net_{o1}}}{(1+e^{-net_{o1}})^2} = \frac{e^{-net_{o1}+1-1}}{(1+e^{-net_{o1}})^2} = \frac{1}{1+e^{-net_{o1}}} - \frac{1}{(1+e^{-net_{o1}})^2}$

$= out_{o1}(1- out_{o1})$

- $\frac{\partial out_{o1}}{\partial net_{o1}} = out_{o1}(1- out_{o1}) = 0{,}7513(1\text{-}0{,}7513) = 0{,}1865$

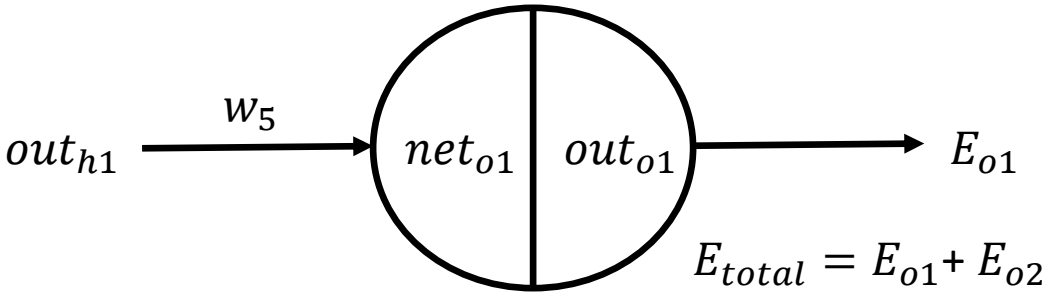
Geri Yayılım Algoritması



- $\frac{\partial net_{o1}}{\partial w_5} = ?$
- $net_{o1} = w_5 * out_{h1} + w_6 * out_{h2} + b_2 * 1$
- $\frac{\partial net_{o1}}{\partial w_5} = out_{h1} = 0{,}5932$

Geri Yayılım Algoritması

- $\frac{\partial E_{total}}{\partial w_5} = \frac{\partial net_{o1}}{\partial w_5} \frac{\partial out_{o1}}{\partial net_{o1}} \frac{\partial E_{total}}{\partial out_{o1}}$



= 0,5932 * 0,1865 * 0,7413 = 0,082

- $\frac{\partial E_{total}}{\partial w_5} = (out_{o1} - target_{o1})\ out_{o1}\ (1 - out_{o1})\ out_{h1}$

- $\frac{\partial E_{total}}{\partial net_{o1}} = \frac{\partial E_{total}}{\partial out_{o1}} \frac{\partial out_{o1}}{\partial net_{o1}} = \delta_{o1} \rightarrow \frac{\partial E_{total}}{\partial w_5} = \delta_{o1}\ out_{h1}$

- $w_5^{+} = w_5 - \eta \frac{\partial E_{total}}{\partial w_5} = 0,4 - 0,5*0,082 = 0,3589$

YSA’nın genel özellikleri

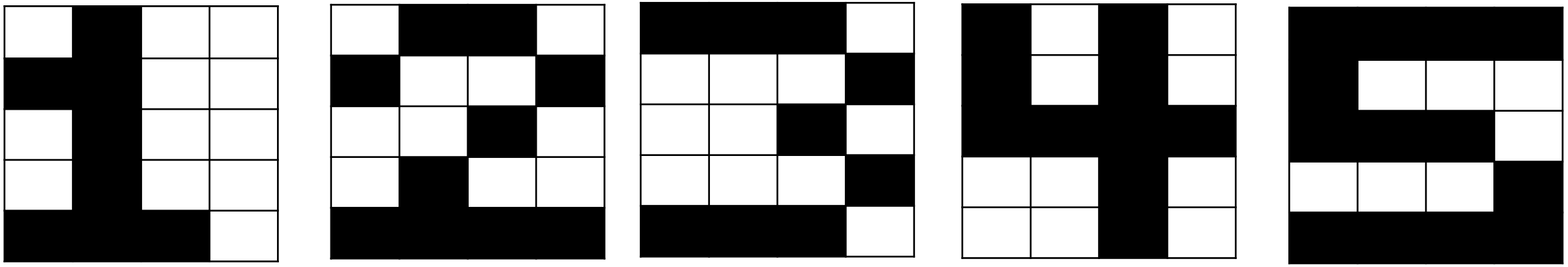
- YSA’lar öğrenmenin dijital ortamdaki bir modelidir.
- YSA’dan bir sonuç elde etmek için ağ, uygun örnekler ile eğitilmelidir.
- YSA eksik ve hatalı bilgilere karşı dayanıklıdır.
- Algoritmik bir çözüm olmayan durumlarda yaygın olarak kullanılır ve başarılı sonuçlar üretir.

YSA’nın olumsuz yönleri

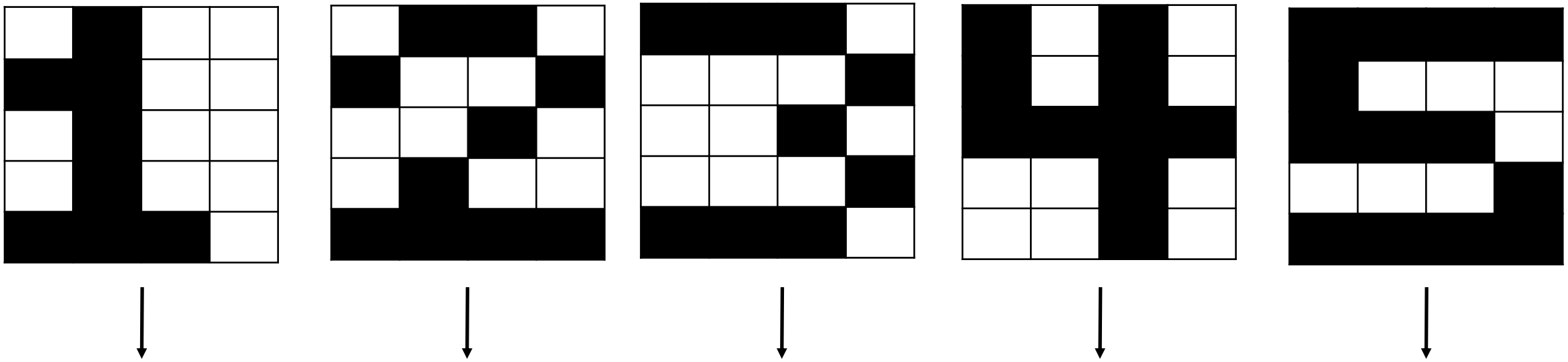
- YSA’lar yerel çözümlere takılabilir.
- Uygun ağ yapısı deneme yanılma yöntemi ile bulunur.
- Gizli katmandaki nöron sayısı uygulamanın hızını önemli derecede etkilemektedir: Örneğin karmaşık görüntülerin analizi için az sayıda nöron içeren ağ yeterli olmazken; çok fazla nöron içeren gizli katman ise işlem hızını büyük ölçüde azaltır.
- YSA eğitiminde eğitim uzayı büyük önem taşımaktadır. Uygun seçilmeyen örnekler ile başarım düşer.

YSA örnek problem

- Yapay sinir ağları ile karakter algılama örneği
  - Elle yazılmış karakterlerin otomatik olarak algılanması uzun zamandır üzerinde çalışılan bir konudur.
  - Karakter tanıma YSA ile yapılabilmektedir. Örneğimizde 0 ile 9 arasındaki karakterler ağa öğretilmektedir.
  - Öncelikle karakterler aşağıdaki gibi matrissel olarak temsil edilir.



YSA örnek problem



0	1	0	0
1	1	0	0
0	1	0	0
0	1	0	0
1	1	1	0

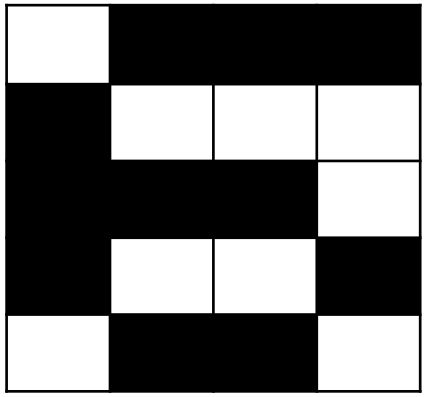
0	1	1	0
1	0	0	1
0	0	1	0
0	1	0	0
1	1	1	1

1	1	1	0
0	0	0	1
0	0	1	0
0	0	0	1
1	1	1	0

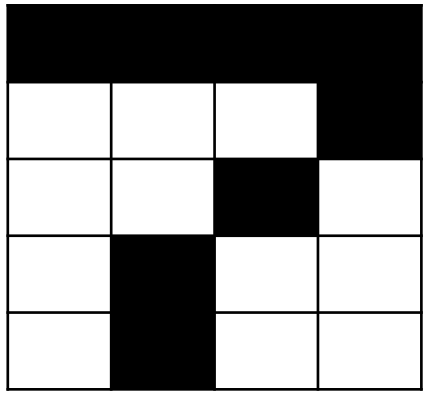
1	0	1	0
1	0	1	0
1	1	1	1
0	0	1	0
0	0	1	0

1	1	1	1
1	0	0	0
1	1	1	0
0	0	0	1
1	1	1	1

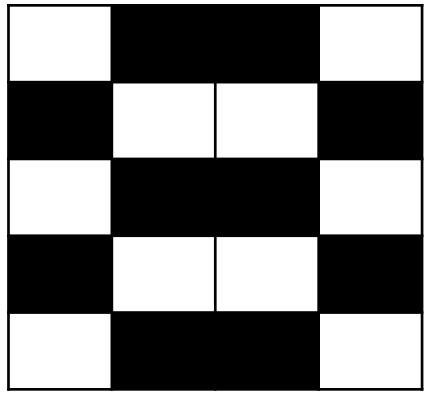
YSA örnek problem



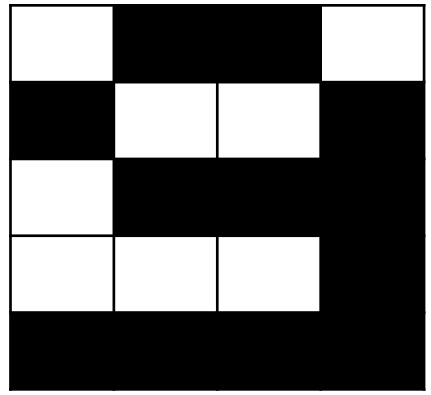
0	1	1	1
1	0	0	0
1	1	1	0
1	0	0	1
0	1	1	0



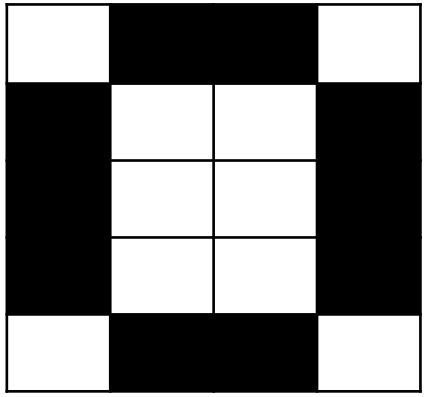
1	1	1	1
0	0	0	1
0	0	1	0
0	1	0	0
0	1	0	0



0	1	1	0
1	0	0	1
0	1	1	0
1	0	0	1
0	1	1	0



0	1	1	0
1	0	0	1
0	1	1	1
0	0	0	1
1	1	1	1



0	1	1	0
1	0	0	1
1	0	0	1
1	0	0	1
0	1	1	0

YSA örnek problem

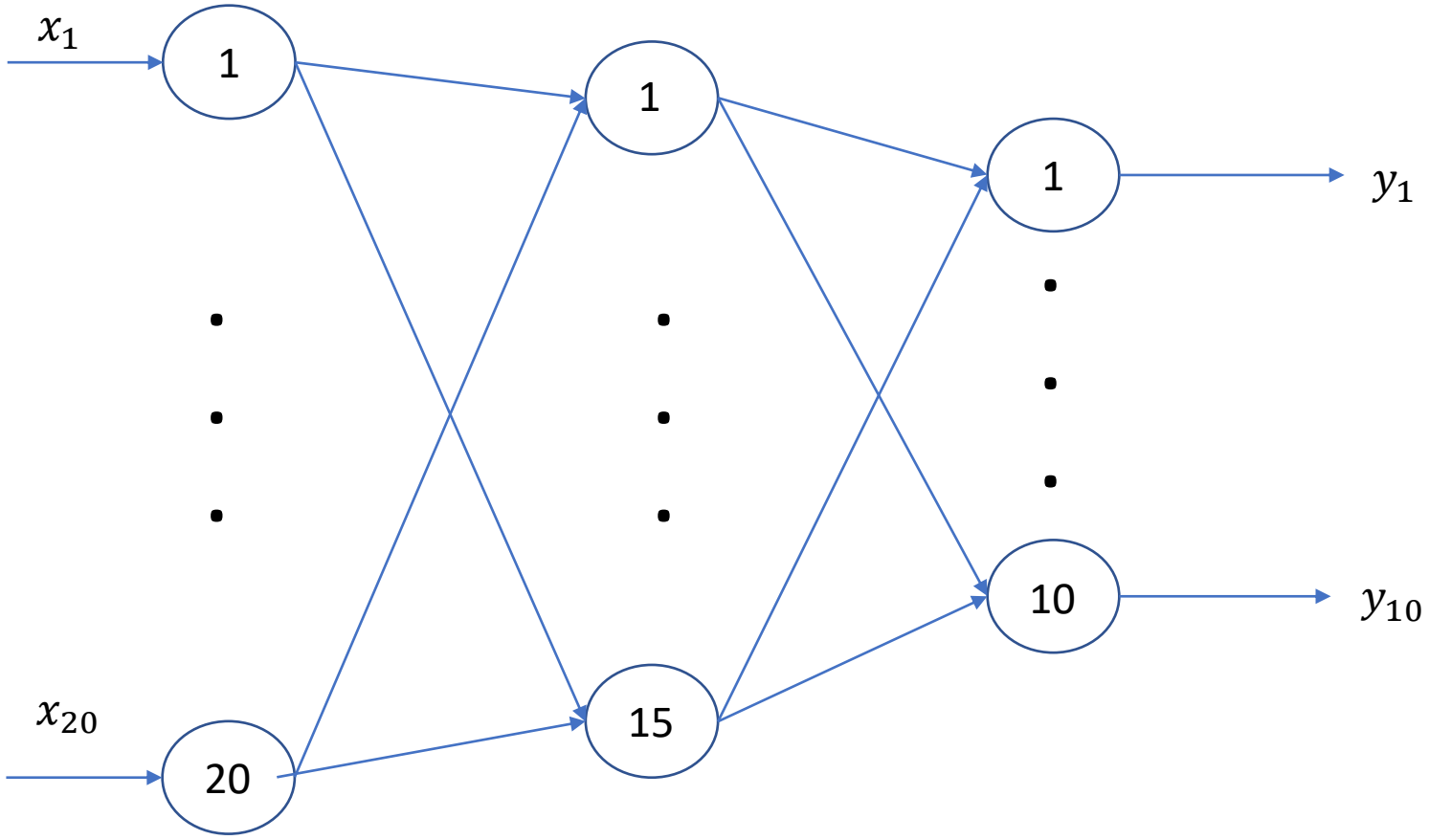
- Giriş karakterleri 4x5 lik matrislerle gösterildiği için ağın giriş katmanındaki sinir sayısı 20 olmalıdır.
- Matrislerdeki bilgiler 1. satırdan başlamak üzere her bir hücredeki değer giriş katmanının her bir sinirine gelecek şekilde ağa uygulanır.
- Çıkış temsili 10 bitlik ikili sayılarla olacak şekilde tasarım yapılacaktır. Bu da çıkış katmanındaki sinir sayısının 10 adet olması anlamına gelir.
- 10 bitlik ikili sayının temsili gösterimi aşağıdaki gibidir.
- Örneğin çıkışta sinirler sırasıyla 1000000000 ürettiyse bu çıkış için sonuç 1'dir.

1	2	3	4	5	6	7	8	9	10	çıkış
1	0	0	0	0	0	0	0	0	0	**1**
0	1	0	0	0	0	0	0	0	0	**2**
0	0	1	0	0	0	0	0	0	0	**3**
0	0	0	1	0	0	0	0	0	0	**4**
0	0	0	0	1	0	0	0	0	0	**5**
0	0	0	0	0	1	0	0	0	0	**6**
0	0	0	0	0	0	1	0	0	0	**7**
0	0	0	0	0	0	0	1	0	0	**8**
0	0	0	0	0	0	0	0	1	0	**9**
0	0	0	0	0	0	0	0	0	1	**0**

YSA örnek problem

- Çıkışlar bu şekilde tasarlandıktan sonra yapılması gereken gizli katmandaki sinir sayısını belirlemektir.
- Gizli katman sinir sayısı genellikle giriş sinir sayısı ile çıkış sinir sayısı arasında bir değer olarak belirlenir.
- Ayrıca öğrenme durumuna göre deneme yanılma yoluyla da bir değer verilebilir.
- Sinirlerin sayısı ne kadar fazla tutulursa ağın eğitimden sonra hatırlama yeteneği o kadar iyi olur, fakat sayının fazla olması eğitim süresini uzatır.
- Bu örnekte gizli katman sayısı 15 olarak belirlenmiştir.

YSA örnek problem



Yandaki ağda;

- Giriş sinir sayısı:20
- Giriş katmanı ile gizli katman arasında 300 bağlantı vardır
- Gizli katman ile çıkış katmanı arasında 150 bağlantı vardır.
- Çıkış sinir sayısı:10
- Örnekte 4x5 lik bir görüntü yerine 224x224 gibi daha büyük boyutlu bir görüntü kullanılsaydı giriş gizli katmanlar ve bunların arasındaki bağlantı sayısı çok büyük sayılara çıkacak ve bu da ağın eğitilmesini çok zorlaştıracaktı.

Ek kaynaklar:

- https://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example/