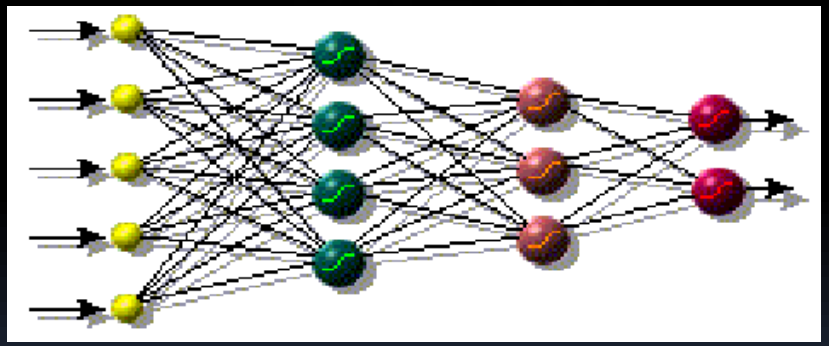
BİLECİK ŞEYH EDEBALİ ÜNİVERSİTESİ

Fen Bilimleri Enstitüsü

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

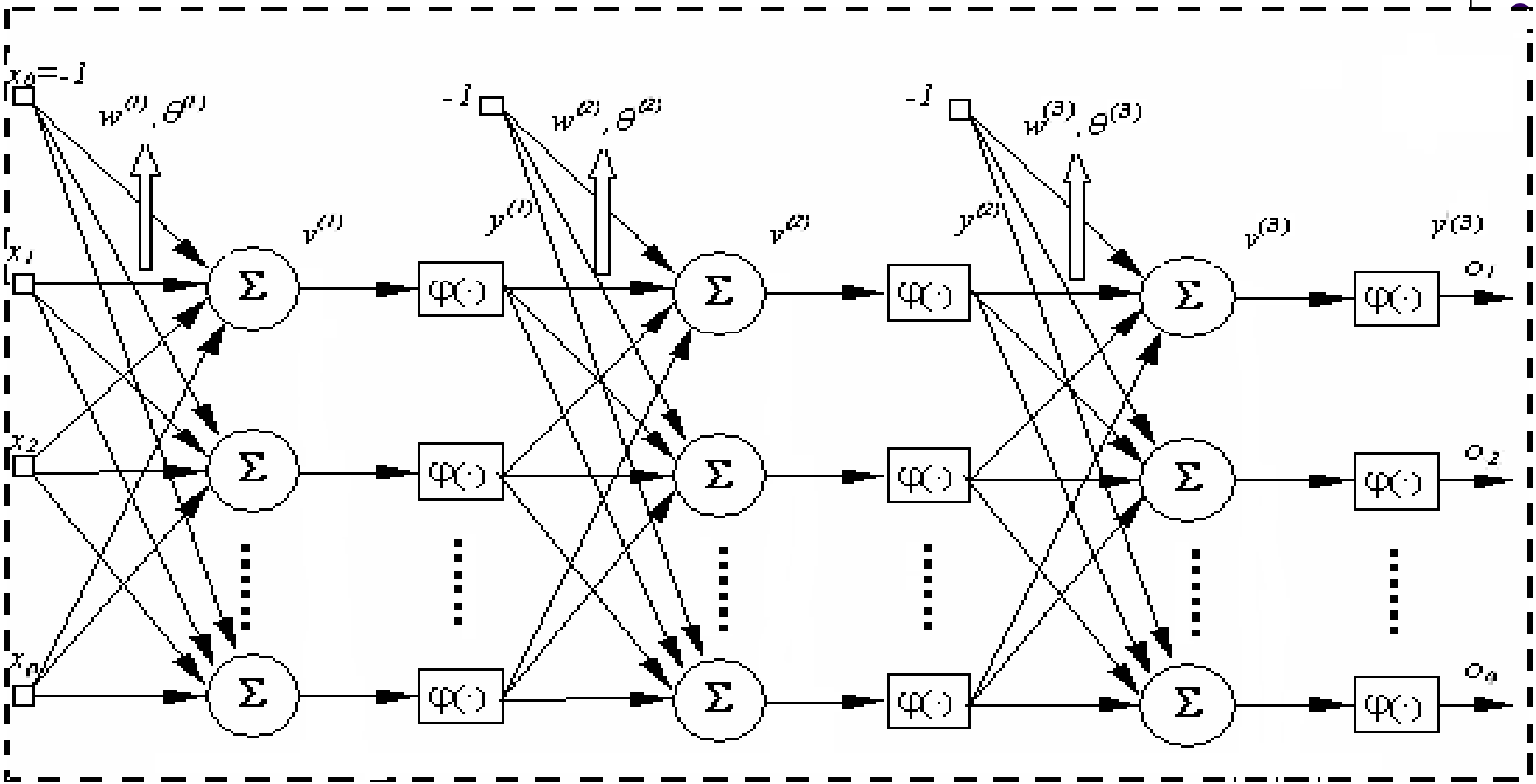
YAPAY SİNİR AĞLARI
BM508



Dr. Cihan KARAKUZU

DERS-2

Çok katmanlı YSA'da İleri Hesaplama

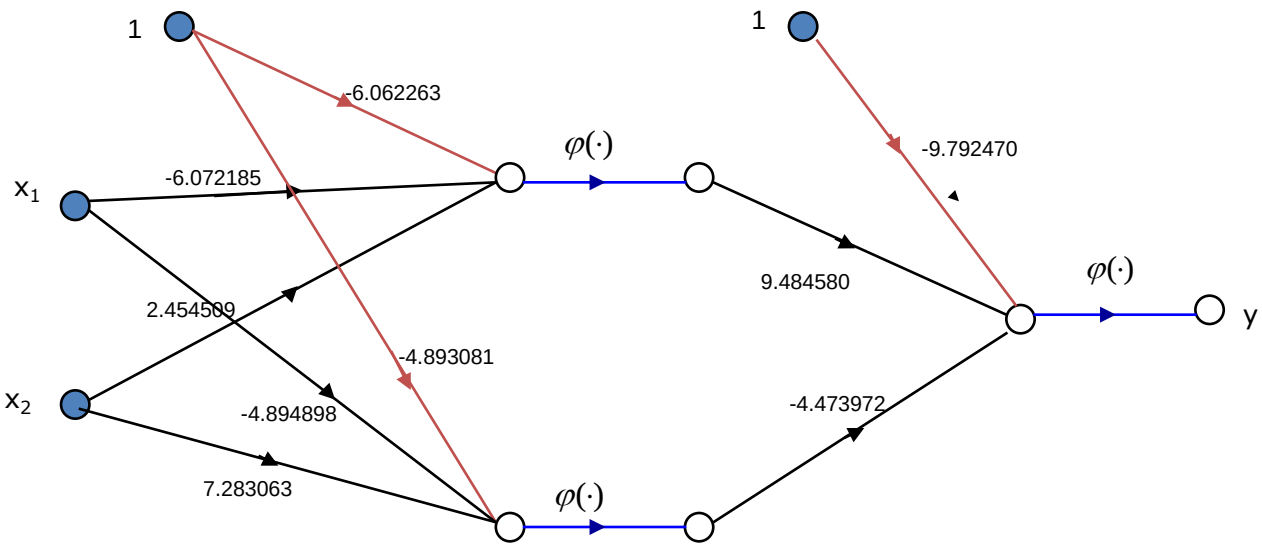


$$x = \begin{bmatrix} x_1 & \cdots & x_n \end{bmatrix}_{1\times n} \quad w^{(1)} = \begin{bmatrix} w_{11} & \cdots & w_{1n} \\ \vdots & \vdots & \vdots \\ w_{h_1 1} & \cdots & w_{h_1 n} \end{bmatrix}_{h_1\times n} \quad \theta^{(1)} = \begin{bmatrix} \theta_1 \\ \vdots \\ \theta_{h_1} \end{bmatrix}_{h_1\times 1} \quad y^{(1)} = \varphi(w^{(1)} x^T + \theta^{(1)})$$

ÇOK KATMANLI YSA'DA İLERİ HESAPLAMA (Devam)

ÖRNEK : *{2,1} yapısında sigmoidal aktivasyon fonksiyonlu 2 katmanlı bir YSA, EX-OR işlevini gerçekleyecek şekilde eğitilmeye aşağıda verilen parametrelerle başlanmıştır. Bu ağın bu parametrelerle EX-OR işlevi için olası tüm girişler için %kaç hatalı çıkış verdiğini belirleyiniz.*

$$W^{(1)} = \begin{bmatrix} -6.072185 & 2.454509 \\ -4.894898 & 7.283063 \end{bmatrix},\ W^{(2)} = \begin{bmatrix} 9.484580 & -4.473972 \end{bmatrix},\ \theta^{(1)} = \begin{bmatrix} -6.062263 \\ -4.893081 \end{bmatrix},\ \theta^{(2)} = \begin{bmatrix} -9.792470 \end{bmatrix}$$



x_1	x_2	y_d	y_a	%e
0	0	0		
0	1	1		
1	0	1		
1	1	0		

[0 1] girişi için bu ağın verilen parametrelerle verdiği çıkışın hesabı aşağıda verilmiştir.

$$v^{(1)} = (W^{(1)} \times X) + \theta^{(1)} = \left(\begin{bmatrix} -6.072185 & 2.454509 \\ -4.894898 & 7.283063 \end{bmatrix} \times \begin{bmatrix} 0 \\ 1 \end{bmatrix} \right) + \begin{bmatrix} -6.062263 \\ -4.893081 \end{bmatrix} = \begin{bmatrix} 2.454509 \\ 7.283063 \end{bmatrix} + \begin{bmatrix} -6.062263 \\ -4.893081 \end{bmatrix} = \begin{bmatrix} -3.607754 \\ 2.389982 \end{bmatrix}$$

$$y^{(1)} = \varphi(v^{(1)}) = \begin{bmatrix} \varphi(-3.607754) \\ \varphi(2.389982) \end{bmatrix} = \begin{bmatrix} \dfrac{1}{1+e^{3.607754}} \\ \dfrac{1}{1+e^{-2.389982}} \end{bmatrix} = \begin{bmatrix} 0.026396 \\ 0.916060 \end{bmatrix}$$

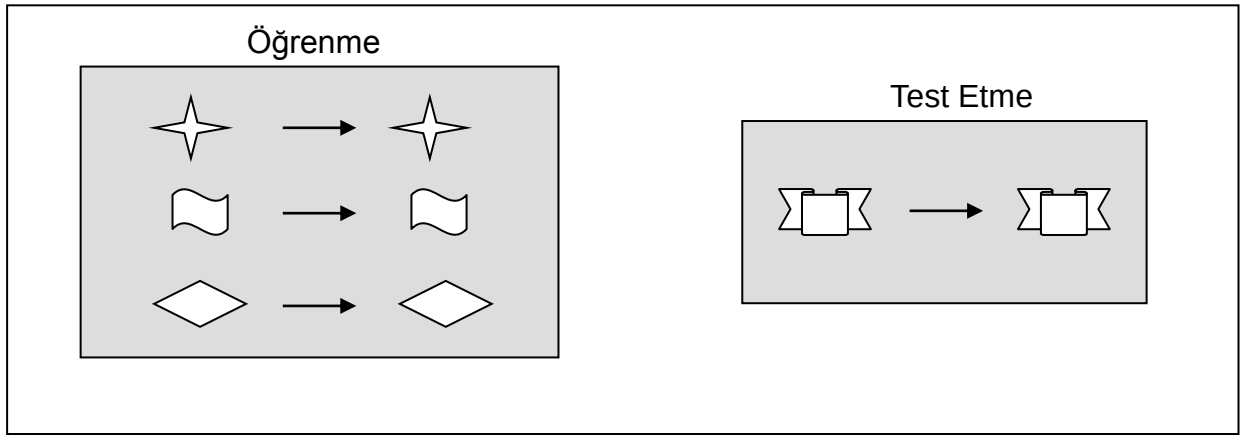
$$v^{(2)} = (W^{(2)} \times y^{(1)}) + \theta^{(2)} = \left(\begin{bmatrix} 9.484580 & -4.473972 \end{bmatrix} \times \begin{bmatrix} 0.026396 \\ 0.916060 \end{bmatrix} \right) + \begin{bmatrix} -9.792470 \end{bmatrix} = -3.848071 - 9.792470 = -13.640541$$

$$y^{(2)} = \varphi(v^{(2)}) = \varphi(-13.640541) = \frac{1}{1+e^{13.640541}} = 1.185x10^{-6} \approx 0$$

Diğer girişler için bu ağın verdiği çıkışları da siz hesaplayın!

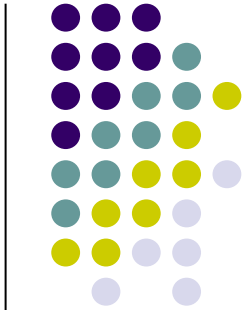
Yapay Sinir Ağlarında Öğrenme ve Test Etme

- YSA'da hücre bağlantı ağırlık ve eşik değerlerinin belirlenmesi işlemine ***"ağın eğitilmesi"*** denir.
- Başlangıçta bu ağırlık ve eşik değerleri rasgele olarak atanır.
- Eğitimde amaç, ağa gösterilen örnekler için doğru çıktıları üretecek ağırlık değerlerini bulmaktır.
- Örnekler ağa defalarca gösterilerek en doğru ağırlık değerleri bulunmaya çalışılır. Ağın doğru ağırlık değerlerine ulaşması örneklerin temsil ettiği olay hakkında genellemeler yapabilme yeteneğine kavuşması demektir. Bu genelleştirme özelliğine kavuşması işlemine ***"ağın öğrenmesi"*** denir.
- Ağ parametre değerlerinin değişmesi belirli kurallara göre yürütülür. Bu kurallara ***"öğrenme kuralları"*** denir.
- Farklı stratejilere göre işleyen değişik öğrenme kuralları geliştirilmiştir.
- YSA'da öğrenmenin iki aşaması vardır. Birinci aşamada ağa gösterilen örnek için ağın üreteceği çıktı belirlenir. Bu çıktı değerinin doğruluk derecesine göre ikinci aşamada ağın bağlantılarının sahip olduğu ağırlıklar değiştirilir.
- Ağın eğitimi tamamlandıktan sonra öğrenip öğrenmediğini (performansını) ölçmek için yapılan denemelere ise ***"ağın test edilmesi"*** denmektedir. Test etmek için ağın eğitim sırasında görmediği örnekler kullanılır. Test etme sırasında ağın ağırlık değerleri değiştirilmez. Test örnekleri ağa gösterilir. Ağ eğitim sırasında belirlenen bağlantı ağırlıklarını kullanarak görmediği bu örnekleri için çıktılar üretir. Elde edilen çıktıların doğruluk değerleri ağın öğrenmesi hakkında bilgiler verir.



- Eğitimde kullanılan örnek setine ***eğitim seti***, test için kullanılan sete ise ***test seti*** adı verilmektedir. Yapay sinir ağlarının bu şekilde bilinen örneklerden belirli bilgileri çıkartarak bilinmeyen örnekler hakkında yorumlar yapabilme (genelleme yapabilme) yeteneğine ***"öğrenme"*** denir.

YSA Temel Öğrenme Kuralları



- **Hebb Kuralı[1949] :**Bu kuralın temeli, "bir nöron diğer bir nörondan giriş alıyorsa ve her iki nöronda aktif ise (matematiksel olarak aynı işarete sahip ise), nöronlar arasındaki ağırlık kuvvetlendirilir" dir.

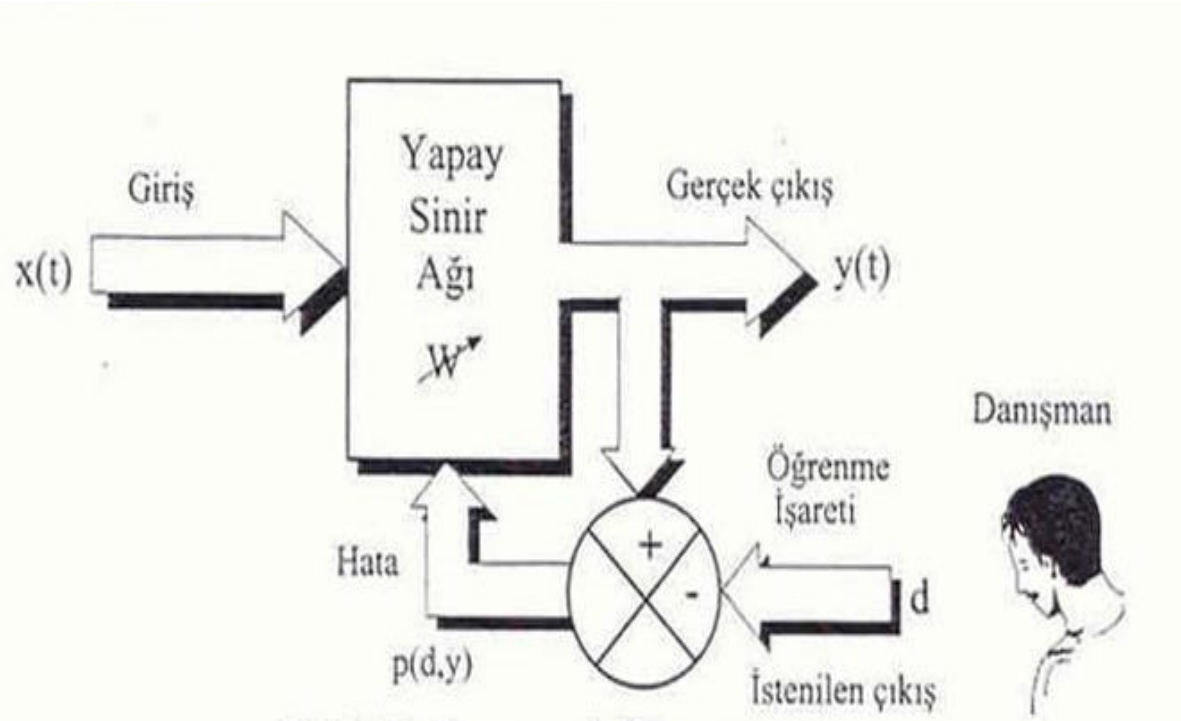
- **Hopfield Kuralı:** Bu kural *zayıflatma* veya *kuvvetlendirme* büyüklüğü dışında Hebb kuralına benzerdir. Eğer istenilen çıkış ve girişin her ikisi aktif veya her ikisi de aktif değilse, öğrenme oranı tarafından bağlantı ağırlığı artırılır, diğer durumlarda ise azaltılır.

- **Delta Kuralı:** Delta kuralı Hebb kuralının değişik bir formudur ve ***en çok kullanılan öğrenme algoritmalarından birisidir***. Bu kural, nöronun gerçek çıkışı ile istenilen çıkış değerleri arasındaki farkı azaltan, giriş bağlantılarını güçlendiren ve sürekli olarak değiştiren bir düşünceye dayanmaktadır. Bu kural, ortalama karesel hatayı bağlantı ağırlık değerlerinin değiştirilmesiyle düşürme prensibine dayanır. Hata, aynı anda bir katmandan bir önceki katmanlara geri yayılarak azaltılır. Bu kural, geri yayılım, Widrow-Hoff veya en küçük ortalama karesel öğrenme kuralı olarak da bilinir.

- **Kohonen Öğrenme Kuralı** :Bu kural, biyolojik sistemlerdeki öğrenmeden esinlenen Kohonen tarafından geliştirilmiştir. Bu kuralda, nöronlar öğrenmek için yarışırlar, kazanan nöronun ağırlıklarını güncellenir. Bu kural "kazanan tamamını alır (winner takes all)" olarak da bilinir. En büyük çıkışa sahip işlemci nöron kazanır, bu nöron komşularını uyarma ve yasaklama kapasitesine sahiptir. Kohonen kuralı hedef çıkışa gereksinim duymadığı için danışmansız bir öğrenme metodudur.

Öğrenme Algoritmalarının Sınıflandırılması

1. Danışmanlı/Eğiticili Öğrenme (Supervised Learning)



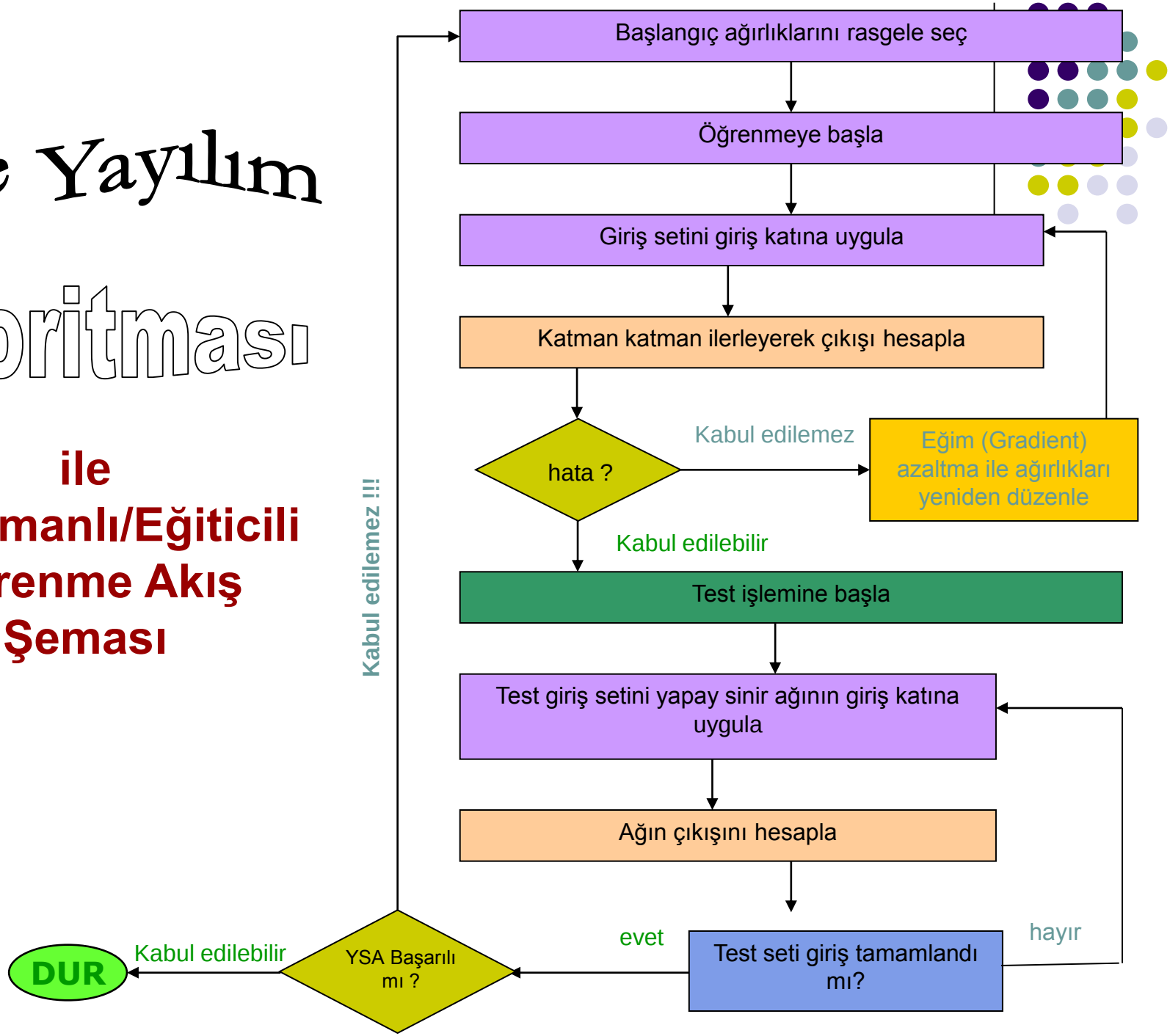
•İstenilen veya arzu edilen çıkış ile gerçek çıkış (ağ çıkışı) arasındaki farka (hataya) göre, nöronlar arası bağlantıların ağırlığı, en uygun çıkışı elde etmek için bir öğrenme algoritmasıyla düzenlenir.

•Widro-Hoff tarafından geliştirilen delta kuralı ve Rumelhart ve McClelland tarafından geliştirilen genelleştirilmiş delta kuralı veya geri yayılım (backpropagation) algoritması danışmanlı öğrenme algoritmalarına örnek olarak verilebilir.

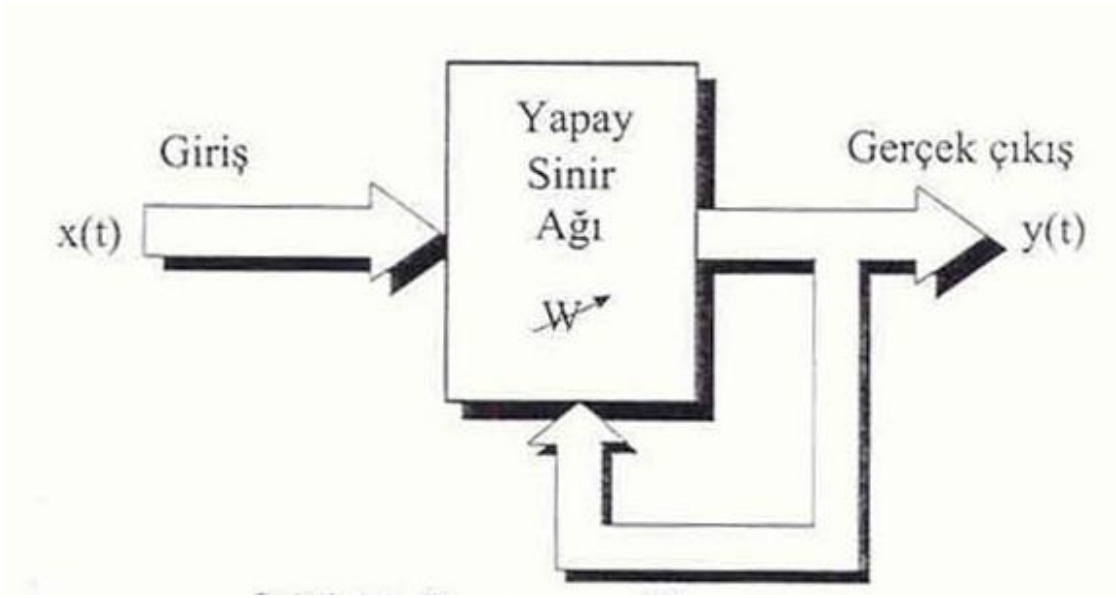
Bu tip öğrenmede, öğrenme aşamasında YSA' ya ne öğrenmesi gerektiği örnek bir çıkışla bildirilir.

Bu tip öğrenme yaklaşımında mutlaka bir danışmana veya YSA' ya ne öğrenmesi gerektiğini aktaracak bir yaklaşıma ihtiyaç vardır.

Geriye Yayılım Algoritması ile Danışmanlı/Eğiticili Öğrenme Akış Şeması



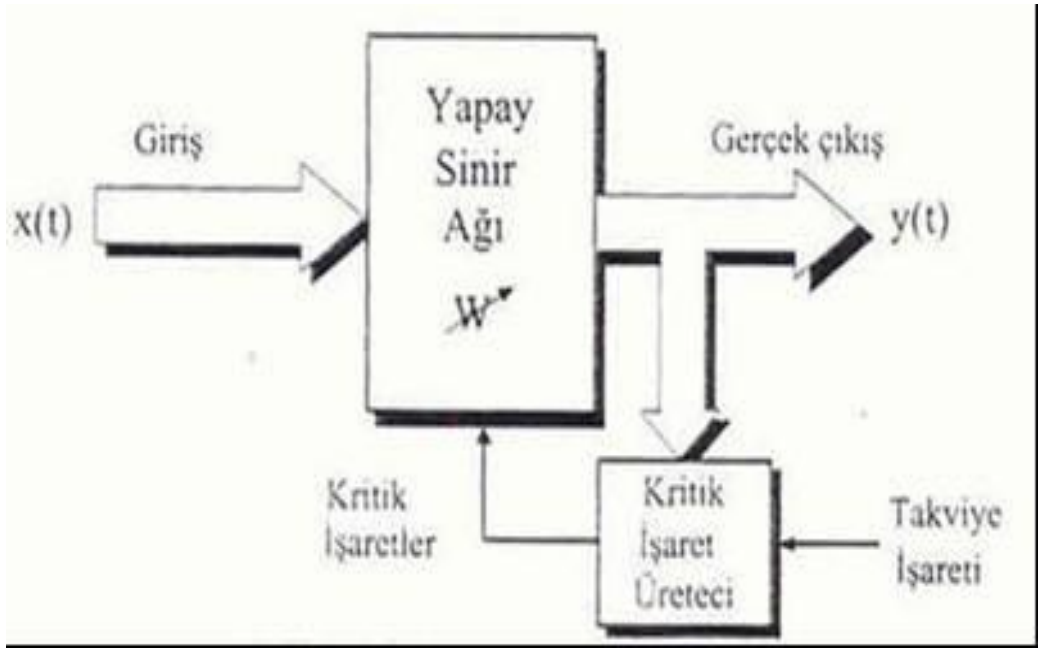
2. Danışmansız/Eğiticisiz Öğrenme (Unsupervised Learning)



Özellikle sınıflandırma problemlerinde kullanılır.

- Bu öğrenme algoritmalarında, istenilen çıkış değerinin bilinmesine gerek yoktur. Öğrenme süresince sadece giriş bilgileri ağa uygulanır. Uygulanan girişe göre, bu giriş verileri arasındaki matematiksel ilişkilere göre bağlantı ağırlıkları ayarlanır.
- Girişe verilen örnekten elde edilen çıkış bilgisine göre, ağ *sınıflandırma* kurallarını kendi kendine geliştirmektedir.
- Grossberg tarafından geliştirilen ART (Adaptive Rezonance Theory) veya Kohonen tarafından geliştirilen Öz-düzenlemeli harita (SOM Self Organizing Map) öğrenme kuralı danışmansız öğrenmeye örnek olarak verilebilir.

3. Takviyeli/Destekli Öğrenme (Reinforcement Learning)



- Bu öğrenme kuralı danışmanlı öğrenme algoritmasının özel bir formudur.
- Giriş değerlerine karşı istenilen çıkış değerlerinin bilinmesine gerek yoktur. YSA' ya bir hedef verilmemekte fakat elde edilen çıkışın verilen girişe karşılık uygunluğunu değerlendiren bir kriter kullanılmaktadır

- ***Optimizasyon problemlerini çözmek için Hinton ve Sejnowski' nin geliştirdiği Boltzmann kuralı veya Genetik algoritma (GA) takviyeli öğrenmeye örnek olarak verilebilir.***