




Çok Katmanlı Ağın Çalışma Şekli


 **Örneklerin toplanması:** Ağın çözmesi istenen olay için daha önce gerçekleşmiş örneklerin bulunması adımıdır. Ağın eğitilmesi için örnekler toplandığı gibi (eğitim seti), ağın test edilmesi için de örneklerin (test seti) toplanması gerekmektedir.


 **Ağın topolojik yapısının belirlenmesi:** Öğrenilmesi istenen olay için oluşturulacak olan ağın yapısı belirlenir. Kaç tane girdi ünitesi, kaç tane ara katman, her ara katmanda kaç tane hücre elemanı ve kaç tane çıktı elemanı olması gerektiği belirlenmektedir.

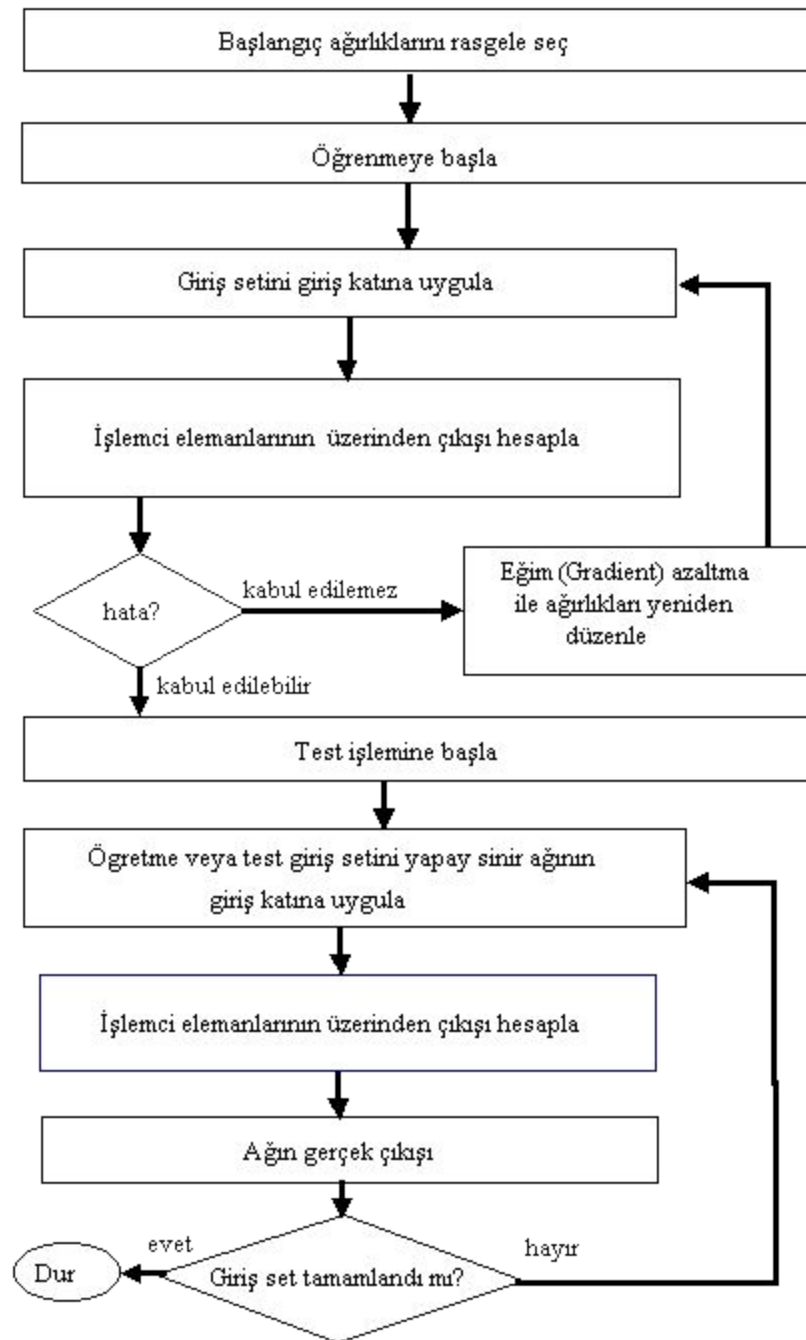
 **Ağın başlangıç değerlerinin atanması:** Hücre elemanlarını birbirine bağlayan ağırlık değerlerinin ve eşik değere başlangıç değerinin atanması

 **Öğrenme setinden örneklerin seçilmesi ve ağa gösterilmesi:** Ağın öğrenmeye başlaması, öğrenme kuralına uygun olarak ağırlıkların değiştirilmesi için ağa örneklerin gösterilmesi.




 **Öğrenme sırasında ileri hesaplamaların yapılması:** Verilen girdi için ağın çıktı değerinin hesaplanması.

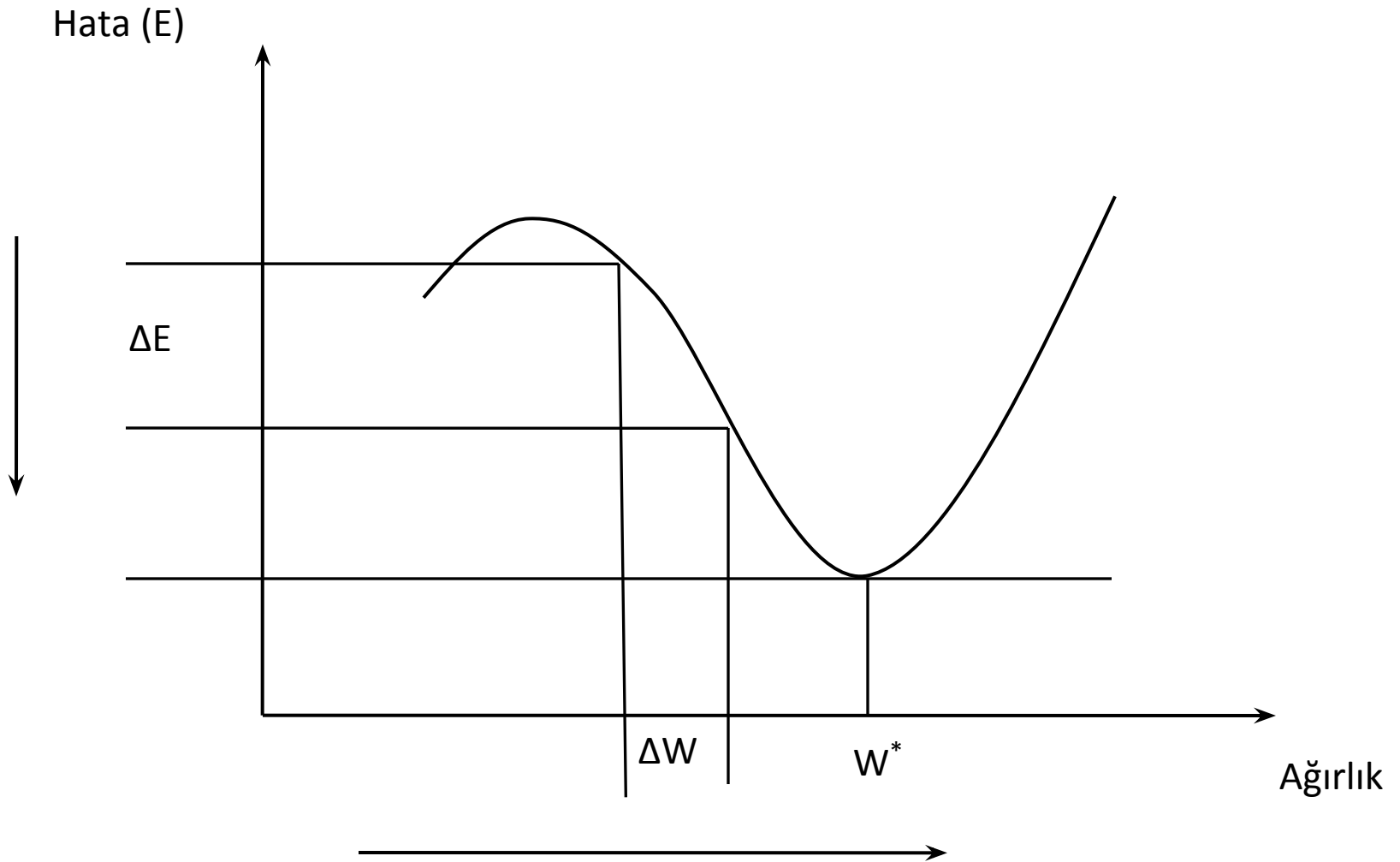
 **Gerçekleşen çıktının beklenen çıktı ile karşılaştırılması:** Ağın ürettiği hata değerlerinin hesaplanması.

 **Ağırlıkların değiştirilmesi:** Geri hesaplama yöntemi uygulanarak üretilen hatanın azalması için ağırlıkların değiştirilmesi.



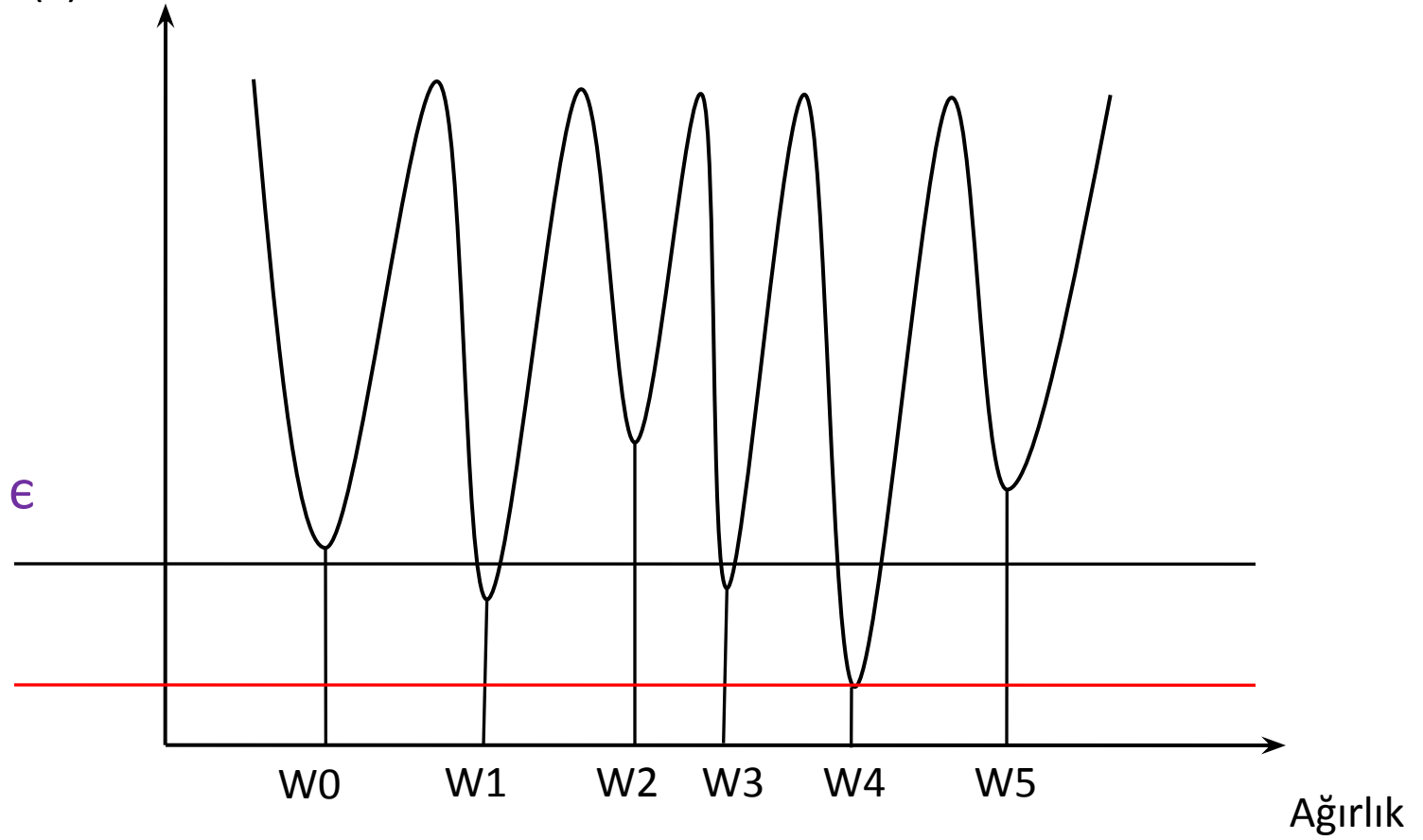
Ağın Eğitilmesi

-  Ağın kendisine gösterilen girdi örneği için beklenen çıktıyı üretmesini sağlayacak ağırlık değerleri bulunmaktadır. Başlangıçta bu değerler rasgele atanmakta ve ağı örnekleri gösterdikçe ağın ağırlıkları değiştirilerek zaman içerisinde istenilen değerlere ulaşması sağlanmaktadır.
-  İstenilen ağırlık değerlerinin ne olduğu ise bilinmemektedir. Bu nedenle yapay sinir ağlarının davranışlarını yorumlamak ve açıklamak mümkün olmamaktadır.
-  Bunun temel nedeni ise, bilginin ağ üzerinde dağıtılmış olması ve ağırlık değerlerinin kendi başlarına herhangi bir anlam göstermemeleridir.





Ağın W^* değerine ulaşması istenmektedir. Her iterasyonda ΔW kadar değişim yaparak hata düzeyinde ΔE kadar bir hatanın düşmesi sağlanır.

Hata (E)



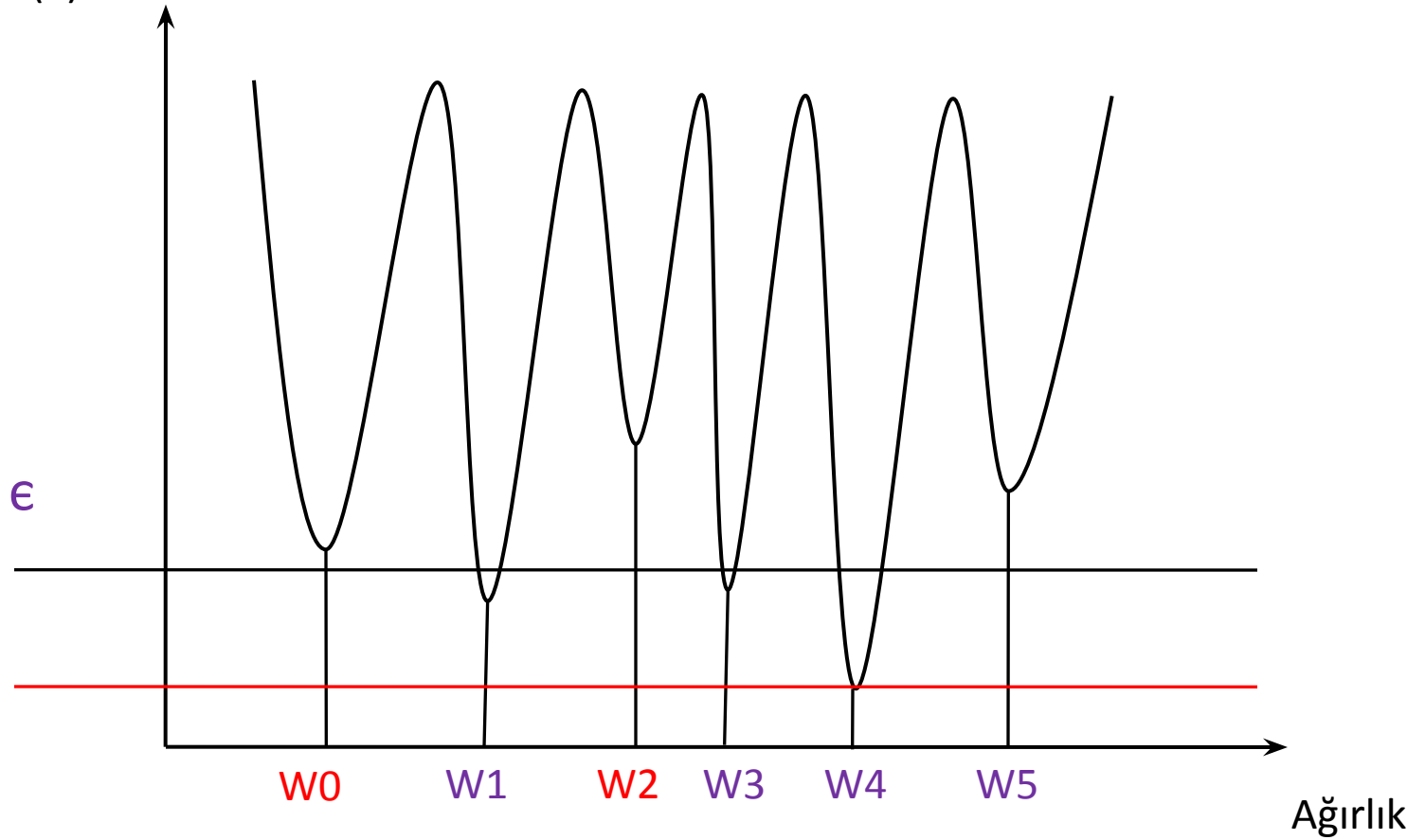
Problemin çözümü için en az hatayı veren ağırlık vektörü W^* olmasına rağmen pratikte bu hata değerini yakalamak mümkün olmayabilir. Bu çözüm ağırlık vektörü sahip olabileceği en iyi çözümdür.

 Bazen farklı bir çözüme takılabilmekte ve performansı daha iyileştirmek mümkün olmamaktadır. Bu nedenle kullanıcılar ağların performanslarında ϵ kadar hatayı kabul etmektedirler. Bu tolerans değerinin altındaki herhangi bir noktada olay öğrenilmiş kabul edilir.

 En iyi çözümün bulunamamasının nedeni aşağıdakilerden biri veya birkaçı olabilir:

- Problem eğitilirken bulunan örnekler problem uzayını %100 temsil etmeyebilir.
- Oluşturulan çok katmanlı ağ için doğru parametreler seçilmemiş olabilir.
- Ağın ağırlıkları başlangıçta tam istenildiği şekilde belirlenmemiş olabilir.
- Ağın topolojisi yetersiz seçilmiş olabilir.

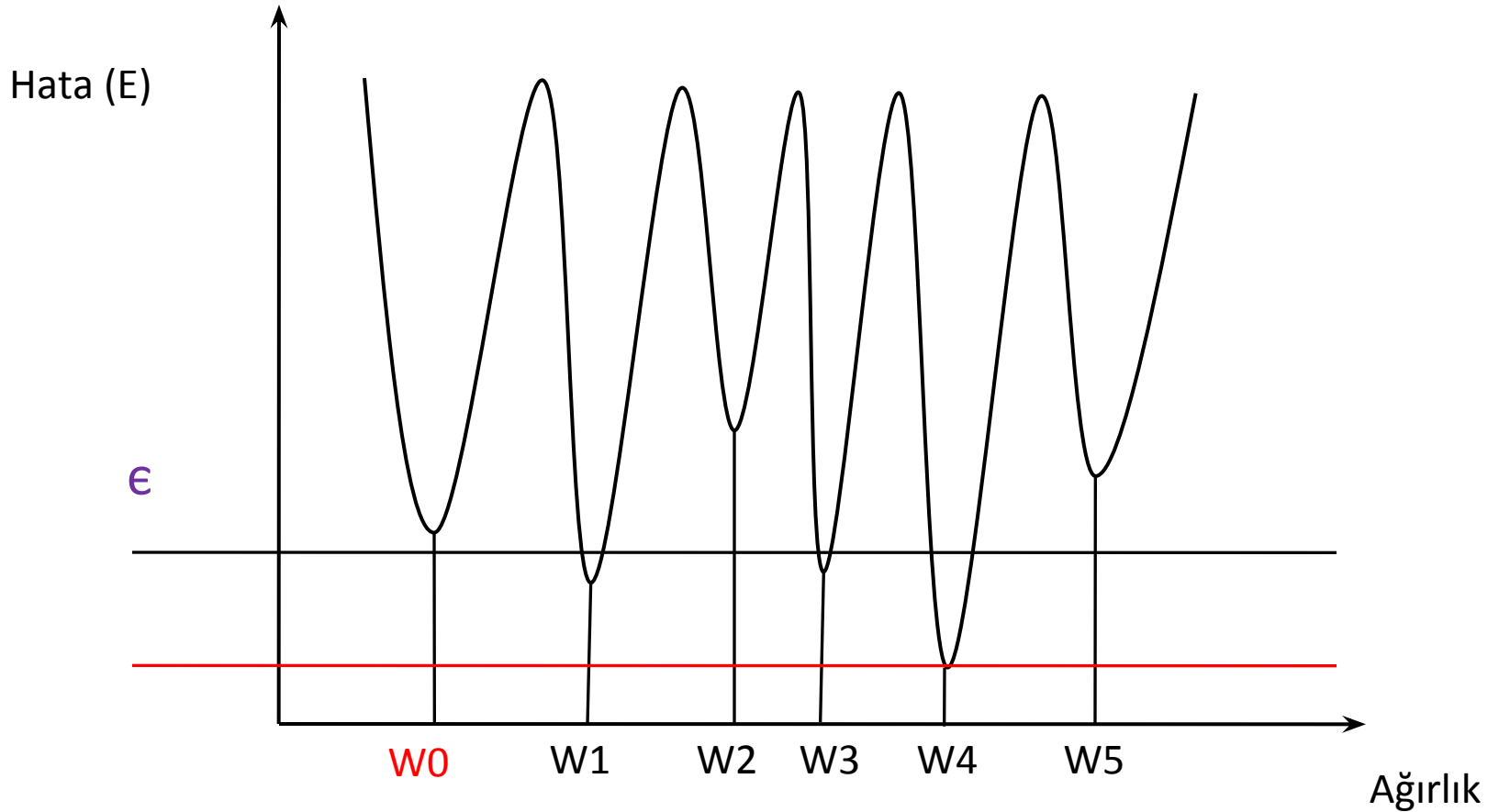
Hata (E)



$W0, W2$ yerel çözümler olup tolerans değerinin üzerinde kaldığı için kabul edilemezler. Yine $W1, W3, W4$ de yerel çözümler olup kabul edilebilirler. Fakat en iyi çözüm $W4$ 'dir. Fakat her zaman bu çözüme ulaşmak az önce sayılan nedenlerden ötürü mümkün olmayabilir.



Bazı durumlarda ağın takıldığı yerel sonuç kabul edilebilir hata düzeyinin üstünde kalabilir. Örneğin aşağıdaki şekilde w_0 ağırlığının bulunması ve hatanın daha fazla azaltılmasının mümkün olmaması gibi.







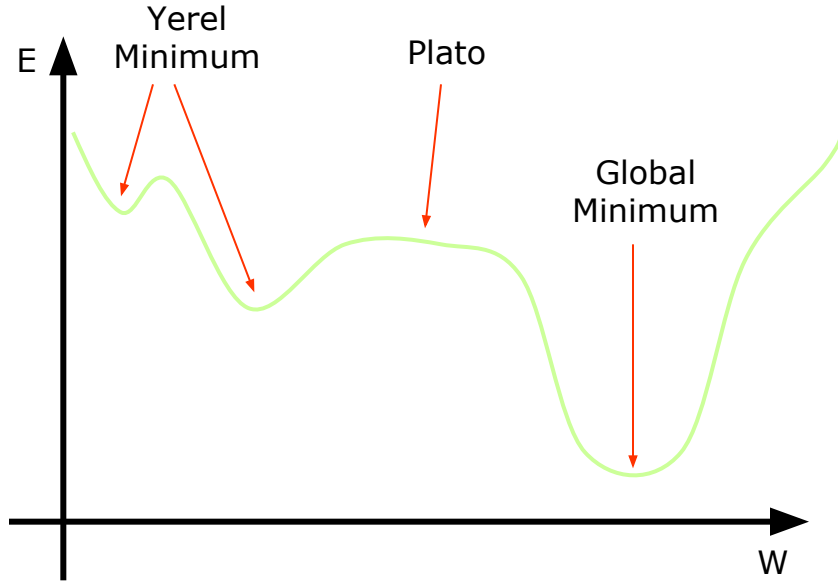
Bu durumda ağın olayı öğrenmesi için bazı değişiklikler yapılarak yeniden eğitilmesi gerekmektedir. Bu değişiklikler şunlar olabilir:

- Başka başlangıç değerleri kullanılabilir.
- Topolojide değişiklikler yapılabilir (Ara katman sayısını arttırmak, proses elemanı sayısını arttırmak veya azaltmak)
- Parametrelerde değişiklik yapılabilir. (Farklı fonksiyonların seçilmesi, öğrenme ve momentum katsayılarının değiştirilmesi)
- Problemin gösterimi ve örneklerin formülasyonu değiştirilerek yeni örnek seti oluşturulabilir.
- Öğrenme setindeki örneklerin sayısı arttırılabilir veya azaltılabilir.
- Öğrenme sürecinde örneklerin ağı gösterilmesi.

Momentum Katsayısı

 Çok katmanlı ağların yerel sonuçlara takılıp kalmaması için momentum katsayısı geliştirilmiştir. Bu katsayının iyi kullanılması yerel çözümleri kabul edilebilir hata düzeyinin altına çekebilmektedir.

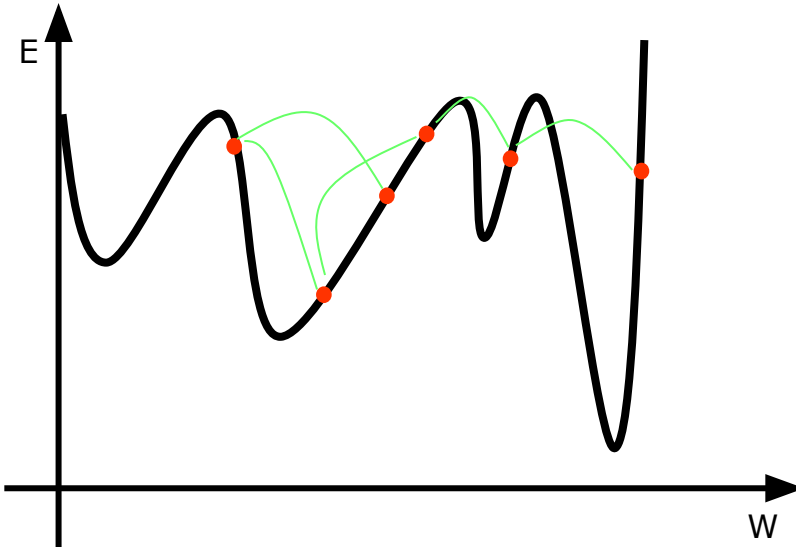
 Çok katmanlı ağların diğer bir sorunu ise öğrenme süresinin çok uzun olmasıdır. Ağırlık değerleri başlangıçta büyük değerler olması durumunda ağın yerel sonuçlara düşmesi ve bir yerel sonuçtan diğerine sıçramasına neden olmaktadır. Eğer ağırlıklar küçük aralıkta seçilirse o zaman da ağırlıkların doğru değerleri bulması uzun sürmektedir.



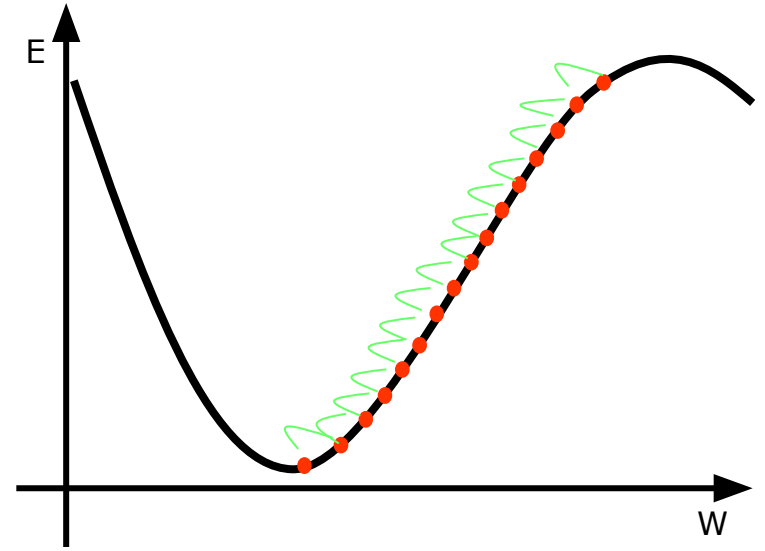
Momentum katsayısı, yerel çözümlere takılmayı önler. Bu değerin çok küçük seçilmesi yerel çözümlerden kurtulmayı zorlaştırır. Değerin çok büyük seçilmesi ise tek bir çözüme ulaşmada sorunlar yaratabilir.

Öğrenme Katsayısı

Öğrenme katsayısı ağırlıkların değişim miktarını belirler.

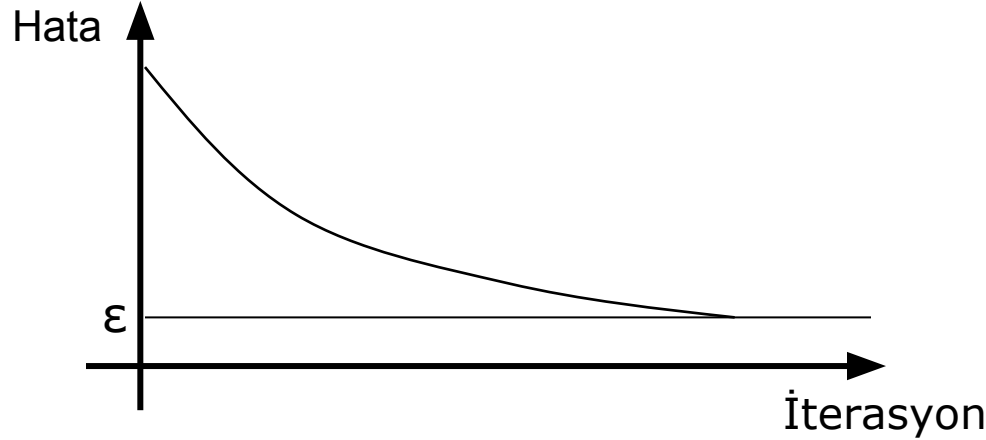


Eğer öğrenme katsayısı gereğinden büyük olursa problem uzayında rasgele gezinme olur. Bunun da ağırlıkları rasgele değiştirmekten farkı olmaz.



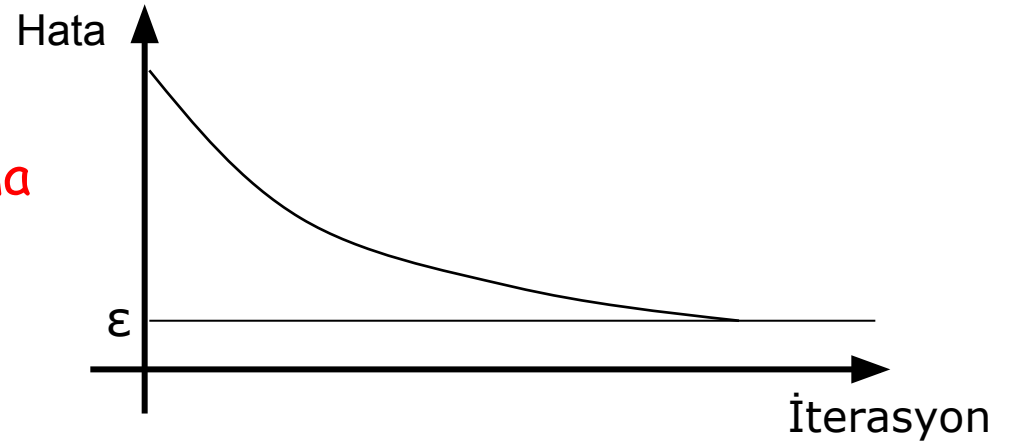
Eğer öğrenme katsayısı çok küçük olursa çözüme ulaşmak daha uzun sürer.

Ağın Hatası

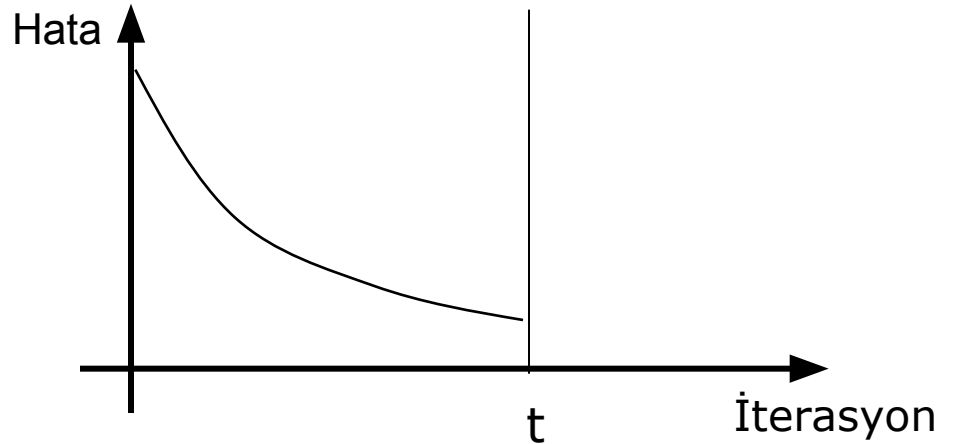


Belirli bir iterasyondan sonra hatanın daha fazla azalmadığı görülür. Bu ağın öğrenmesini durdurduğu ve daha iyi bir sonuç bulunamayacağı anlamına gelir. Eğer elde edilen çözüm kabul edilemez ise o zaman ağ yerel bir çözüme takılmış demektir.

Hatanın belli bir değerin
altına düşmesi sonucu durma



Belirli sayıda iterasyondan
sonra durma




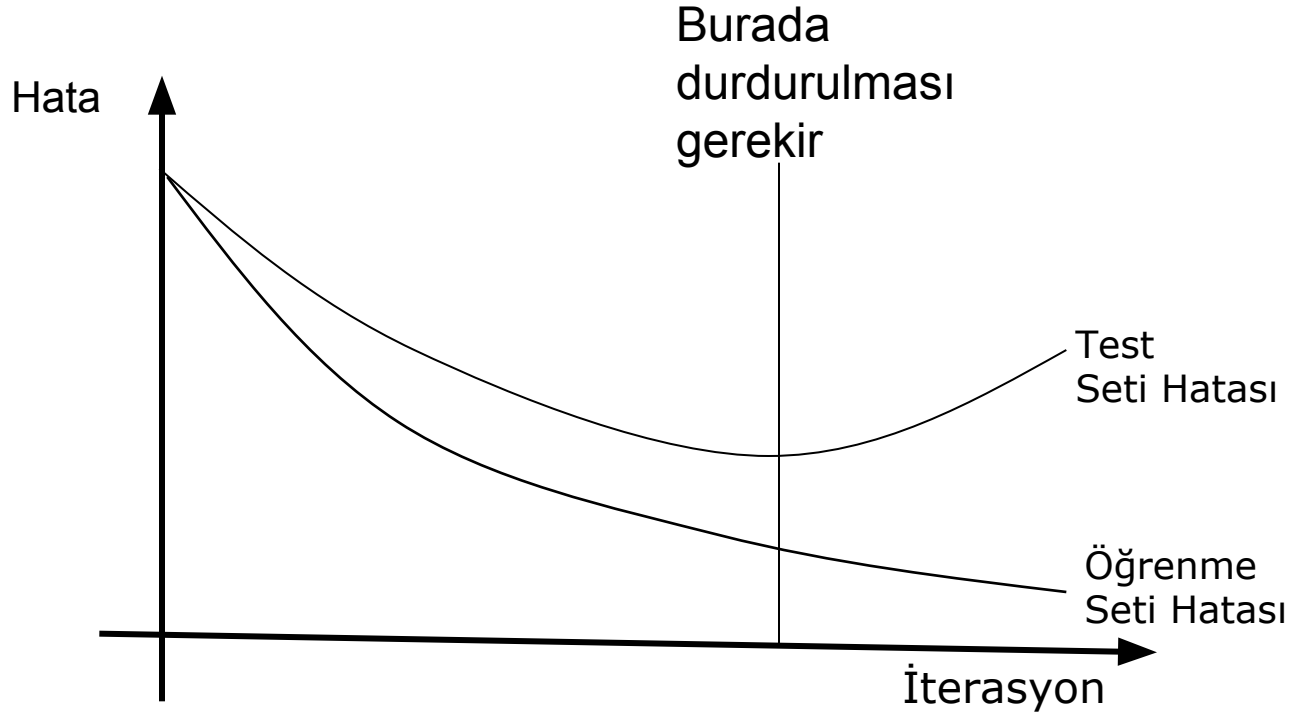
Çok Katmanlı Ağın Performansının Ölçülmesi

- ✓ Bir yapay sinir ağının performansı denilince öğrenme yeteneğinin ölçülmesi anlaşılır. Ağın performansı ağı daha önce hiç görmediği test örnekleri gösterilince bu örnekler karşısında ürettiği doğru cevaplar oranı şeklinde ölçülür.

$$\text{Performans} = \frac{\text{Test setinde bulunan örneklere verilen doğru cevap sayısı}}{\text{Test setinde bulunan toplam örnek sayısı}} \times 100$$

Ağın Ezberlemesi


-  Bazı durumlarda eğitilen ağ eğitim setindeki bütün örneklerle %100 doğru cevap üretmesine rağmen test setindeki örneklerle doğru cevaplar üretememektedir. Bu durumda ağın öğrenmediği fakat öğrenme setini ezberlediği görülmektedir.




Ağ gereğinden fazla eğitilirse problemi öğrenmek yerine verileri ezberler. Bu da ağın genelleme yapamamasını ve hatalı sonuçlar üretmesine neden olur.

Çok Katmanlı Ağlarda Dikkat Edilmesi Gereken Noktalar

- ✓ **Örneklerin seçilmesi:** Seçilen örnekler problem uzayını temsil edebilecek nitelikte olması gerekir. Çok katmanlı ağ tasarımcısı, problem uzayının her bölgesinden ve uzayı temsil eden örnekler seçmesi gerekir.
- ✓ **Girdi ve çıktıların ağa gösterilmesi:** Problem uzayı sayısal verilerden oluşmuyorsa, bu örnekleri sayısal olarak temsil etmek gerekir. Bu dönüştürme çeşitli şekillerde olabilmekte ve bu da ağın performansını etkilemektedir.

 **Girdilerin sayısal gösterimi:** Ağa sunulan girdilerin sayısal duruma dönüştürülmesi her zaman kolay olmamaktadır. Bu da tasarımı zorlaştırabilmektedir.

 **Çıktıların sayısal gösterimi:** Girdilerde olduğu gibi çıktılarda da sayısal gösterim probleminden probleme değişmektedir. Bir problem için birden fazla yöntem kullanılarak sayısal gösterim sağlanabilir. Bunların en iyisinin hangisi olduğu bilinmemektedir. Önemli olan uygun olanı bulmaktır.



Başlangıç değerinin atanması: Başlangıç değerinin atanması performansı etkilemektedir. Genel olarak ağırlıklar belirli aralıkta atanmaktadır.

- Bu ağırlıklar büyük tutulursa ağın yerel çözümler arasında sürekli dolaştığı,
- Küçük olması durumunda ise, öğrenmenin geç gerçekleştiği görülmüştür.
- Tecrübeler, -1.0 ile 0.1 arasındaki değerlerin başarılı sonuçlar ürettiğini göstermektedir.



Öğrenme ve momentum katsayısının belirlenmesi: Öğrenme katsayısı daha önce de belirtildiği gibi ağırlıkların değişim miktarını belirlemektedir.

- Büyük değerler seçilirse, yerel çözümler arasında ağırların dolaşması diğer bir deyişle osilasyon yaşamaması mümkündür.
- Küçük değerler seçilirse öğrenme zamanı artmaktadır.

Momentum katsayısı, bir önceki iterasyon değişiminin belirli bir oranının yeni değişim miktarını etkilemesidir. Bu özellikle yerel çözüme takılan ağırların bir sıçrama ile daha iyi sonuçlar bulmasını sağlamak amacı ile geliştirilmiştir.




Momentum katsayısı, bir önceki iterasyon değişiminin belirli bir oranının yeni değişim miktarını etkilemesidir. Bu özellikle yerel çözüme takılan ağların bir sıçrama ile daha iyi sonuçlar bulmasını sağlamak amacı ile geliştirilmiştir.

- Değerin küçük olması yerel çözümlerden kurtulmayı zorlaştırır.
- Çok büyük değerler ise bir çözüme ulaşmada sorunlar yaşanabilir.



Örneklerin ağa sunulması: İki farklı yaklaşım söz konusudur.

- i. **Sıralı Sunum:** Örnek setindeki bütün örneklerin ağa gösterilme şansı eşittir.
- ii. **Rasgele Sunum:**
 - Seçilen bir örnek tekrar set içerisine dahil edilip rasgele seçim yapılır. Örneklerin ağa gösterimi eşit değildir.
 - Rasgele seçilen örnek eğitim setine tekrar dahil edilmez. Örneklerin ağa gösterilme şansı eşittir.

 **Ağırlıkların değiştirilme zamanı:** İki farklı yaklaşım söz konusudur. Ağırlıkların değiştirilmesi öğrenme kuralına göre yapılmaktadır. Genel olarak 3 durumda ağırlıkların değiştirilmesine izin verilmektedir.

- Her örnek ağa gösterildiğinde
- Belirli sayıda örnek gösterildiğinde
- Bütün örnek seti gösterildiğinde


 **Girdi ve Çıktıların ölçeklendirilmesi** : Ölçeklendirme değişik şekillerde yapılmaktadır.

i. **Girdilerin Ölçeklendirilmesi**: Öncelikli olarak girdi vektörü normalize edilir.

$$X' = \frac{X}{|X|}$$

Ardından örnekleri oluşturan değerler belirli bir aralık içerisine çekilirler.

$$X' = \frac{X - X_{\min}}{X_{\max} - X_{\min}}$$

 **Çıktıların ölçeklendirilmesi:** Çıktıların ölçeklendirilmesi için kullanılan yöntemler girdilerin ölçeklendirilmesi için kullanılan yöntemlerle aynı olabilir.

Ağ öğrenme yaptıktan sonra ölçeklendirilmiş çıktı üreteceğinden ağ çıktıları dış dünyaya verilirken orijinal şekle dönüştürülmesi gerekir. Bunun için ölçeklendirme formülünün tersini kullanmak gerekir.

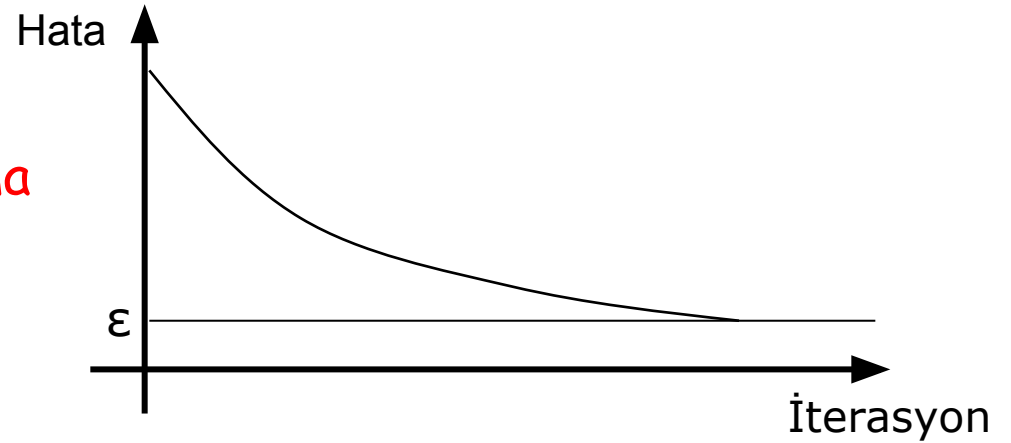
Örneğin, çarpım tablosundaki 2 ile çarpımda

Orijinal veri seti; Girdiler :	$2*2,$	$2*3,$	$2*4,$...
Çıktı :	4,	6,	8,	...
Ölçeklendirilmiş; Girdiler:	$0.2*0.2,$	$0.2*0.3,$	$0.2*0.4,$... 10' a bölünmüş)
Çıktı :	0.04,	0.06,	0.08,	...
Eğitilmiş Ölçeklendirilmiş Çıktı :	$0.2*0.3=0.0598$			
Eğitilmiş Gerçek Çıktı :	5.98 (100 ile çarpılmış)			

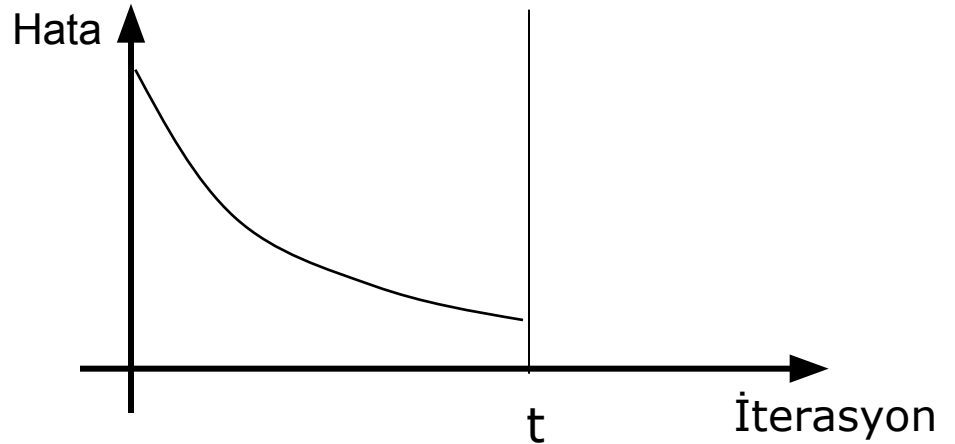


Durdurma kriterinin belirlenmesi:

Hatanın belli bir değerin altına düşmesi sonucu durma



Belirli sayıda iterasyondan sonra durma





Ara katmanların ve her ara katmandaki hücre elemanlarını sayısının belirlenmesi:

Çok katmanlı ağ modelinde herhangi bir problem için kaç tane ara katman ve her ara katmanda kaç tane hücre elemanı kullanılması gerektiğini belirten bir yöntem yoktur.

Bu konudaki çalışmalar deneme yanılma yönteminin etkin olduğunu göstermektedir.

Bazı durumlarda başlangıçta bir ağ oluşturup zaman içinde büyütülerek veya küçültülerek istenilen ağa ulaşılır.



Ağların büyütülmesi veya budanması :

Küçük bir ağdan başlayıp büyük bir ağa doğru eğitim esnasında sürekli hücre elemanlarını arttırmak.

Büyük bir ağdan başlayıp küçük bir ağa doğru eğitim sırasında sürekli ağı küçültmek ve hücre elemanlarını teker teker ağdan çıkarmak.