



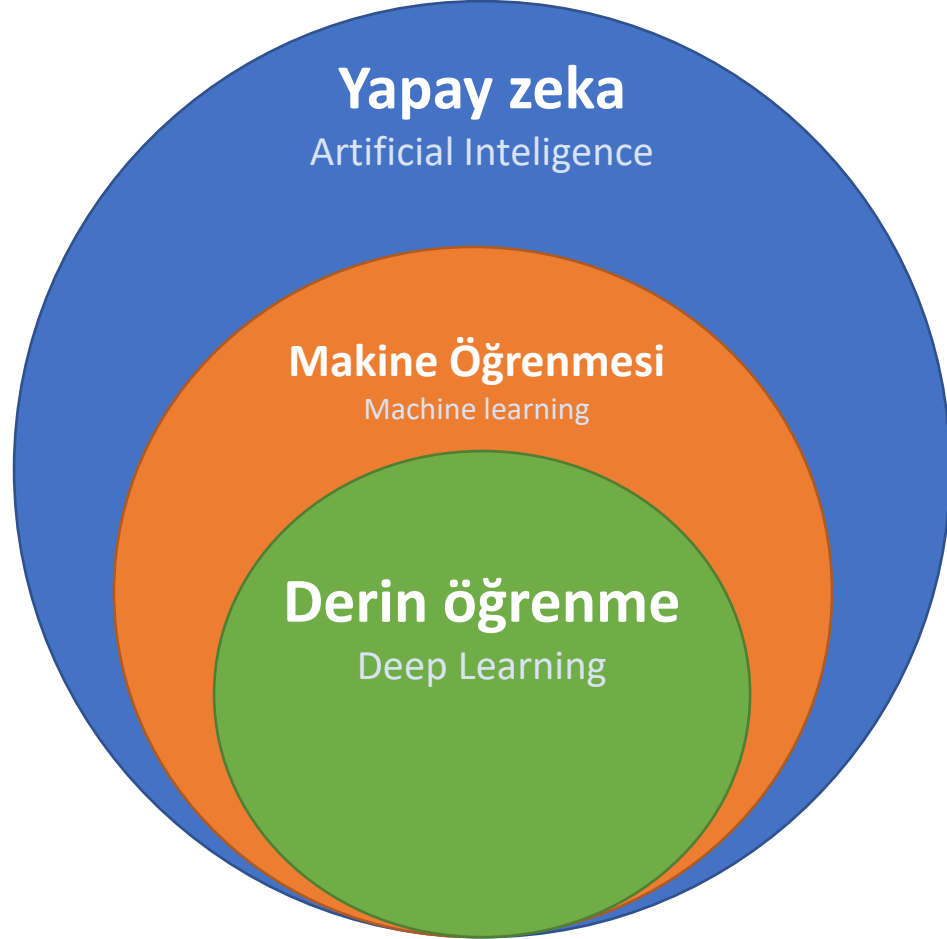
sıfırdan...

Yapay Sinir ağıları

artificial neural networks

from strach....

recep yavuz
turan



Özel Yapay zeka Artificial Narrow Intelligence

İnsan zekâsından
zayıf

Günümüz

Genel Yapay Zeka Artificial Genral Intelligence

İnsan zekâsına eşit

2040

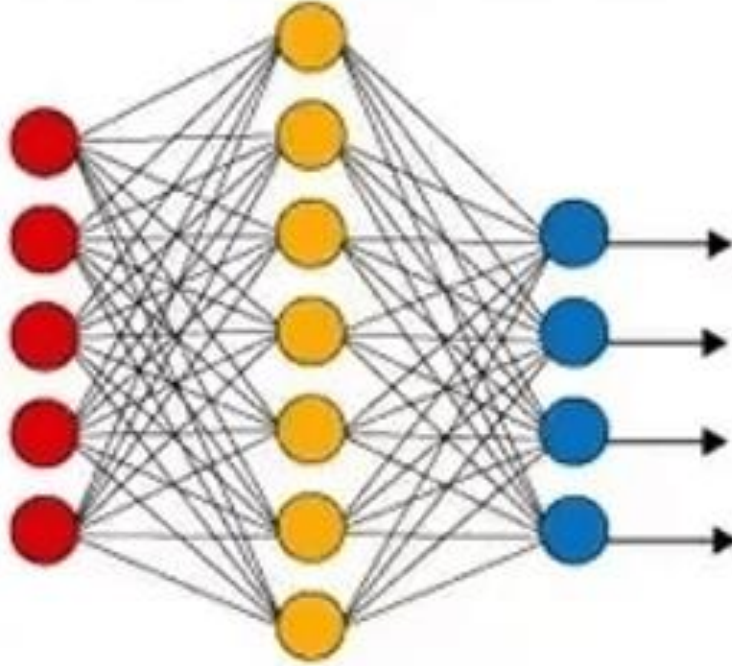
Süper yapay zeka Artificial
Super Intelligence

İnsan zekâsından çok
daha gelişmiş

2060

Basit yapay sinir ağı

Simple Artificial Neural Network



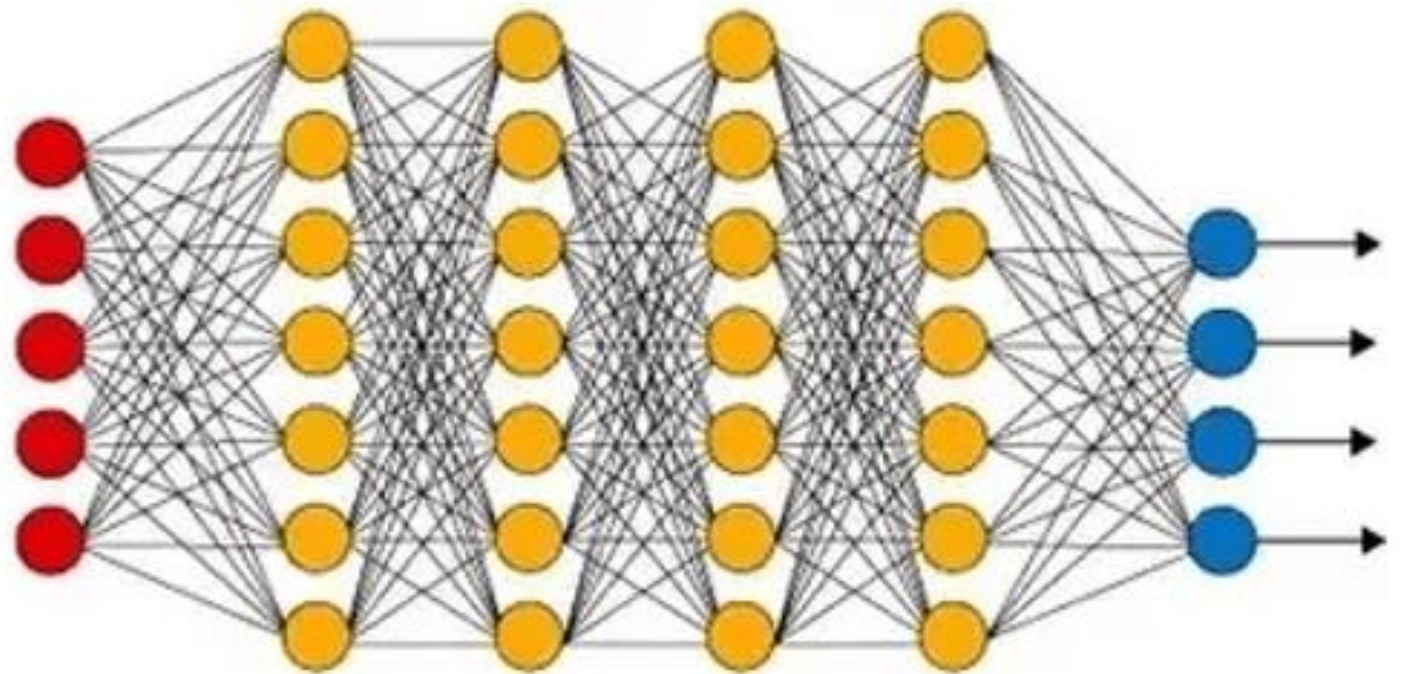
 **Giriş nöronları**
Input Neurons

 **Gizli nöronlar**
Hidden Neurons

 **Çıkış nöronları**
Output Neurons

Derin öğrenme sinir ağı

Deep Learning Neural Network



Yönlü bir graf oluşturacak şekilde yapay nöronların düzenlenmesiyle oluşur....

Sınıflandırma, tahmin gibi işlemleri öğrenip uygulamak için tasarlanan makine öğrenmesi sistemleridir.

Beyinde bulunan biyolojik nöronlardan esinlenilmiştir.

Kontrol, örüntü tanıma, finans, tıp, veri madenciliği, sıralı işlem tanıma, jeolojik bilimler gibi bir çok alanda kullanılmaktadır.



özet

ysa gelişimi progress of ann

brief history

1938

Nicolas Rashevsky

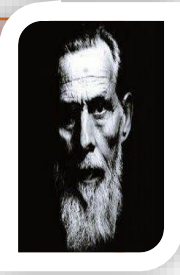
- Nöral Alan teorisi
- Sinir ağlarındaki yayılım ve aktivasyonu diferansiyel denklemlerle ifade etmiştir.



1943

Walter Pitts ve Warren Sturgis McCulloch

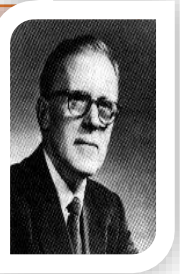
- Basit eşik fonksiyonlarla biyolojik nöronlara benzer yapay modeli sunmuşlardır.



1949

Donald Olding Hebb

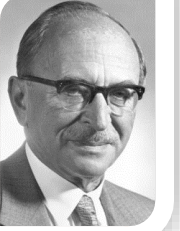
- The Organization of Behavior
- Pavlov'un klasik koşullanmasına dayanan öğrenme teorisini sunmuştur:
«Belirli bir snaps üzerinden sürekli tekrar eden tetiklenme nöronun iletkenliğini artırır.»



1954

Denis Gabor

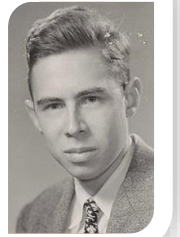
- Gözlenen çıkış işareti ile geçmiş bilgiye dayanarak üretilen işaretin ortalama karesel hatalarını minimize eden optimum ağırlıkları dereceli azalma yöntemiyle tespit eden bir öğrenme filtresi geliştirmiştir.



1958

Frank Rosenblatt

- McCulloch ve Pitts nöron modelini kullanarak bir öğrenme metodu geliştirmiştir.
- «Perceptron»un mucidir.



1960

Bernard Widrow ve Marcian Edward "Ted" Hoff Jr.

- **Adaline** –(Adaptive Linear Neuron /Adaptive Linear Element)
- Dereceli azalma yöntemiyle ortalama karesel hataları minimize ederek öğrenen bir ağ geliştirmişlerdir.



1969

Marvin Minsky ve Seymour Papert

- «Perceptrons» kitabı
- Temel perceptronların sınırlarını hesaplama kabiliyetlerinin kötülüğünü göstermişlerdir.



1982

John Hopfield

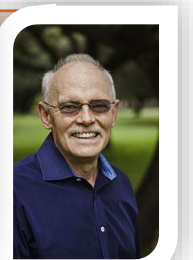
- İlişkisel hafıza olarak kullanılabilecek bir tekrarlayan sinir ağını istatistiksel yöntemlerle açıklamıştır.



1986

David Everett Rumelhart ve James Lloyd "Jay" McClelland

- Çok katmanlı ağların Geri yayılım yöntemiyle öğrenmesi Minsky ve Papert tarafından eleştirilere cevap olmuştur.

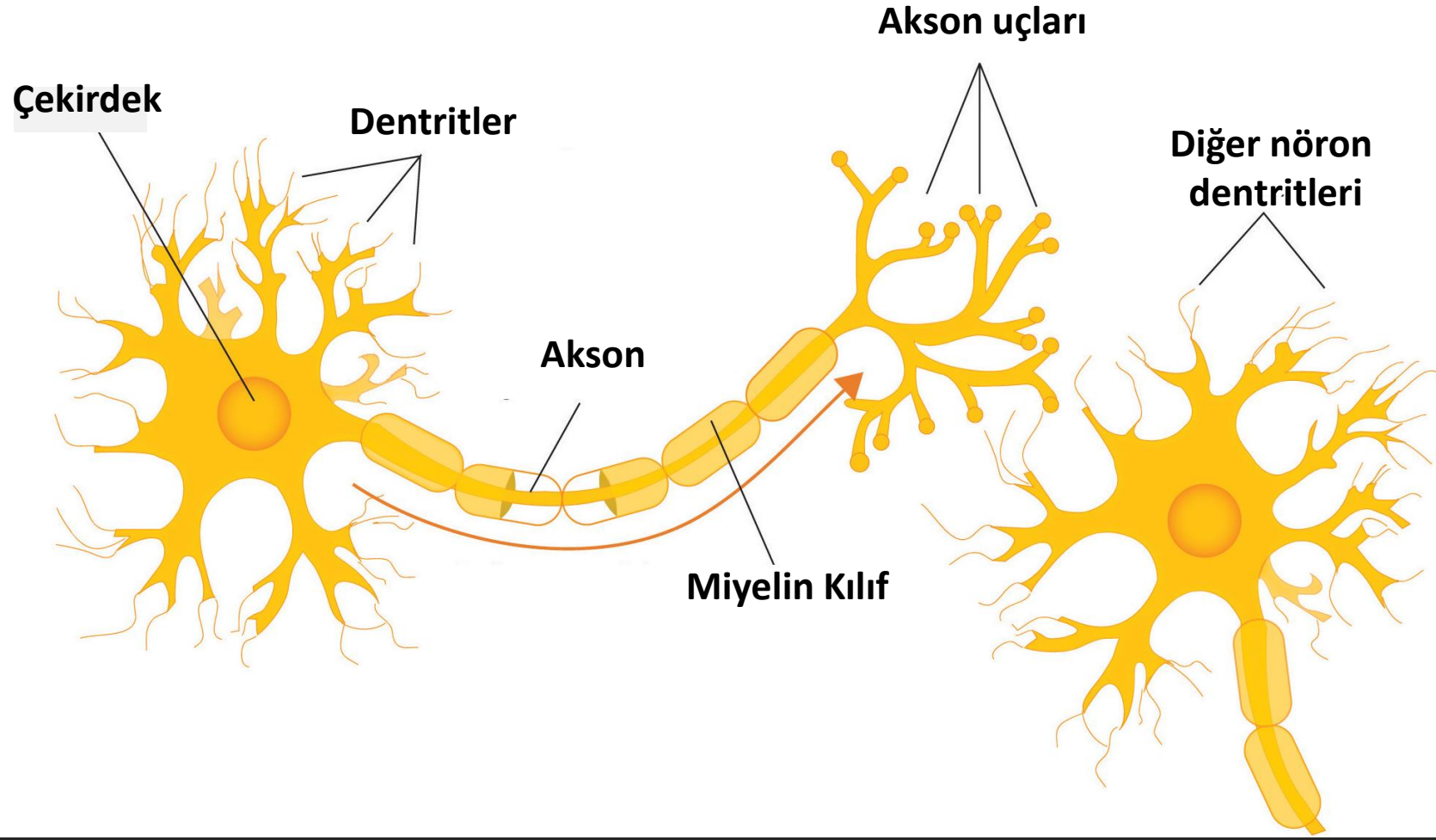


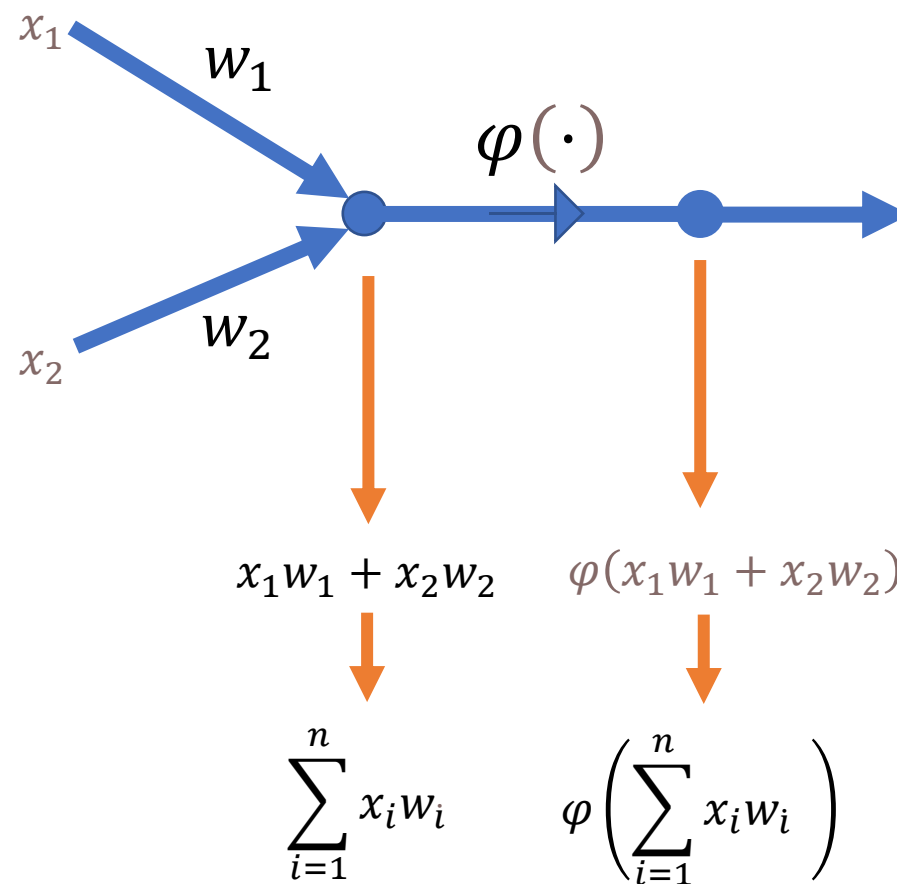
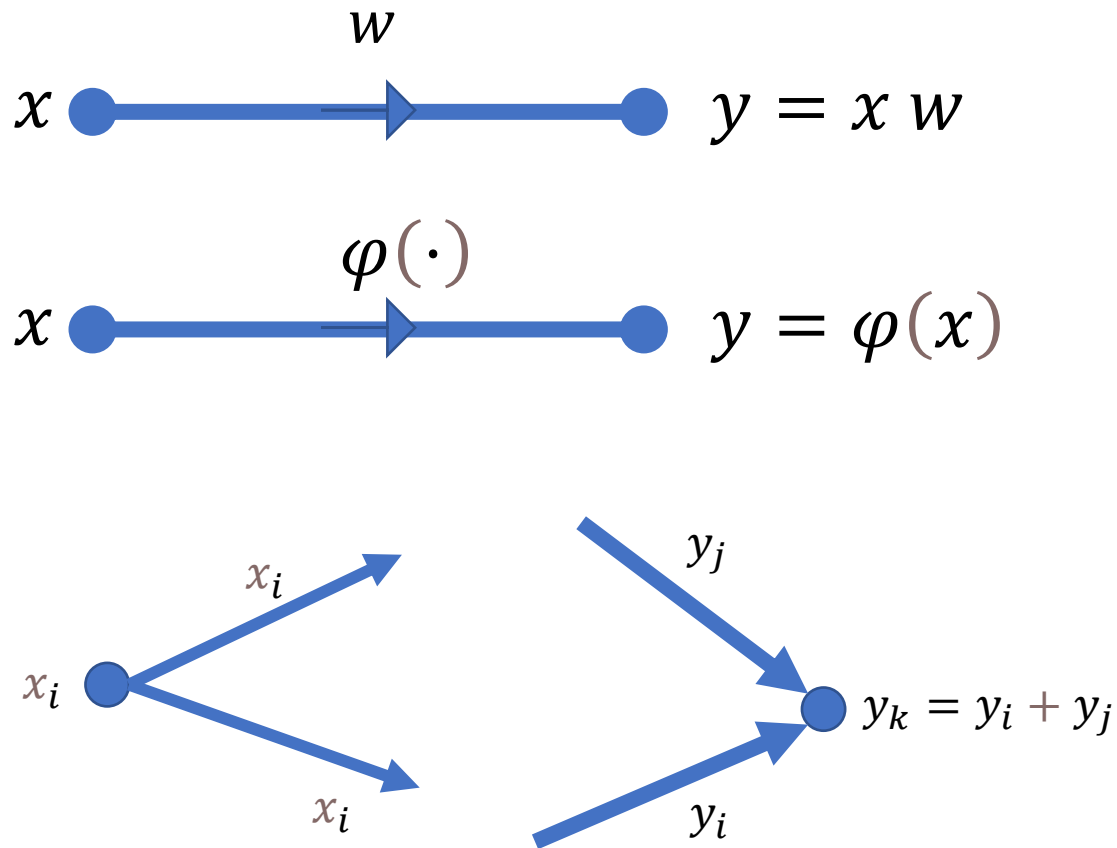


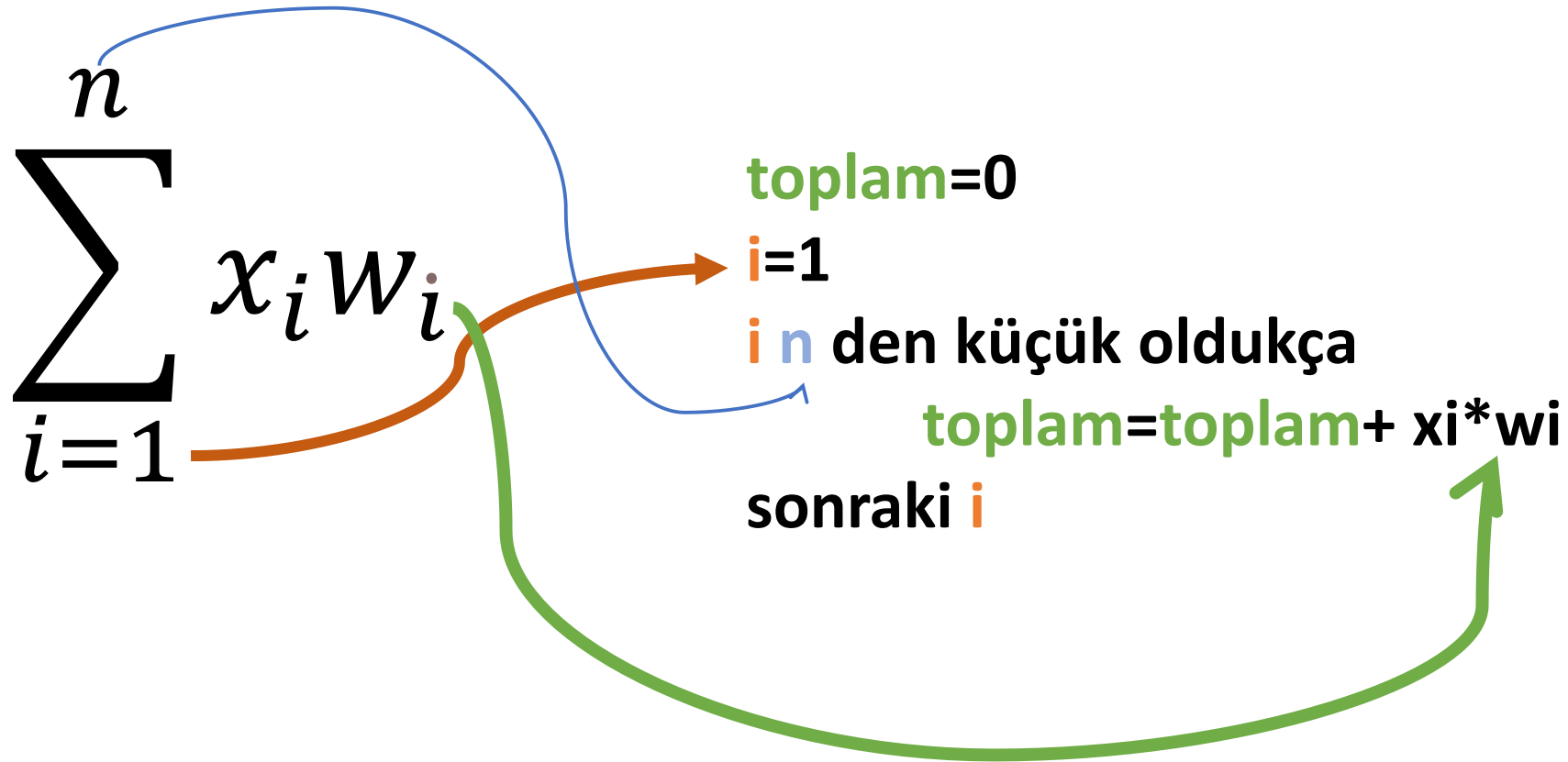
yapay **n**öron**a**rtificial **n**euron

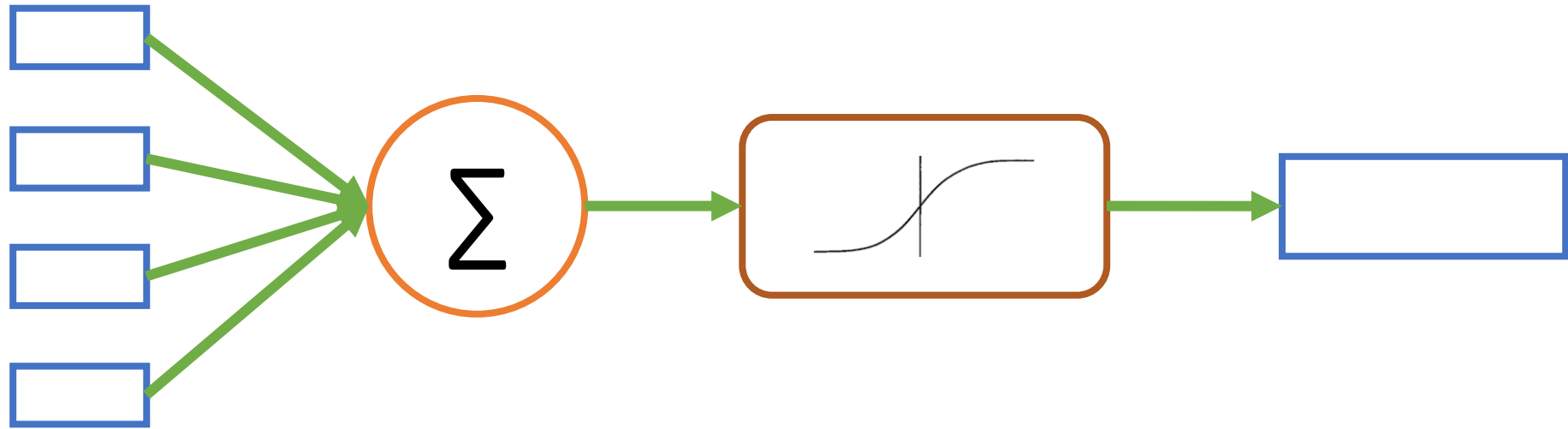
YSA'lar biyolojik nöronlardan esinlenerek yapılmışlardır. Ancak özellikle belirtmek gerekir ki beynin birebir kopyasını oluşturmak gibi bir amaç taşımazlar.

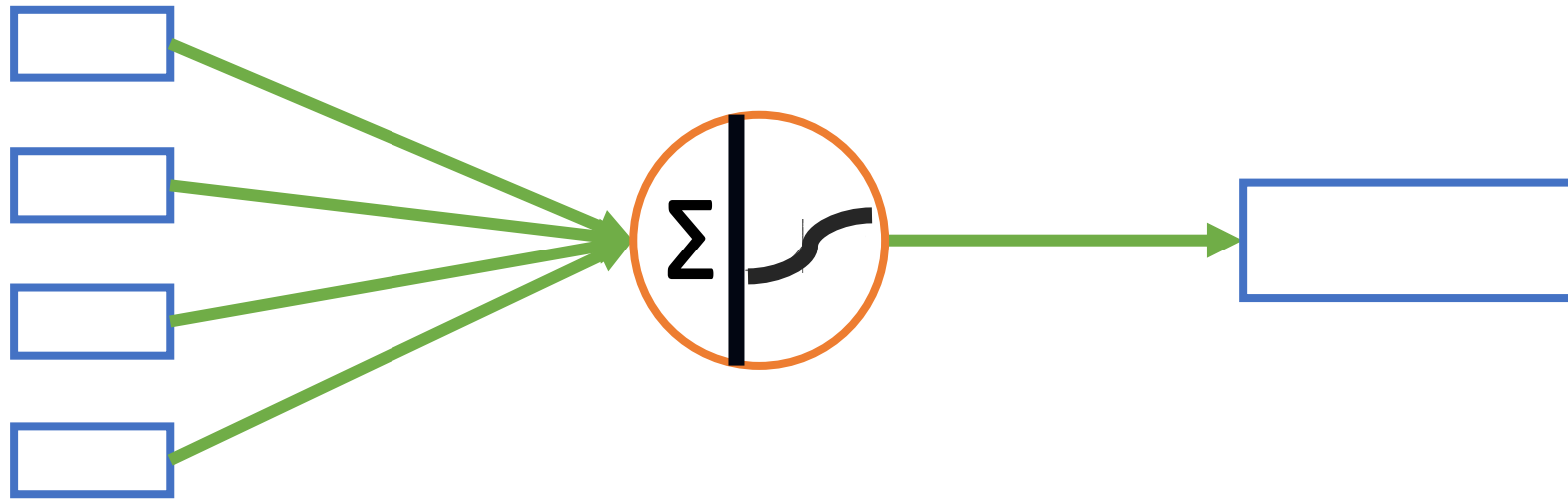
İnsan Beyninin 100 000 000 000 nöron ve 100 000 000 000 000 sinaps(nöronlar arasındaki bağlantı) olduğu belirtilmektedir.

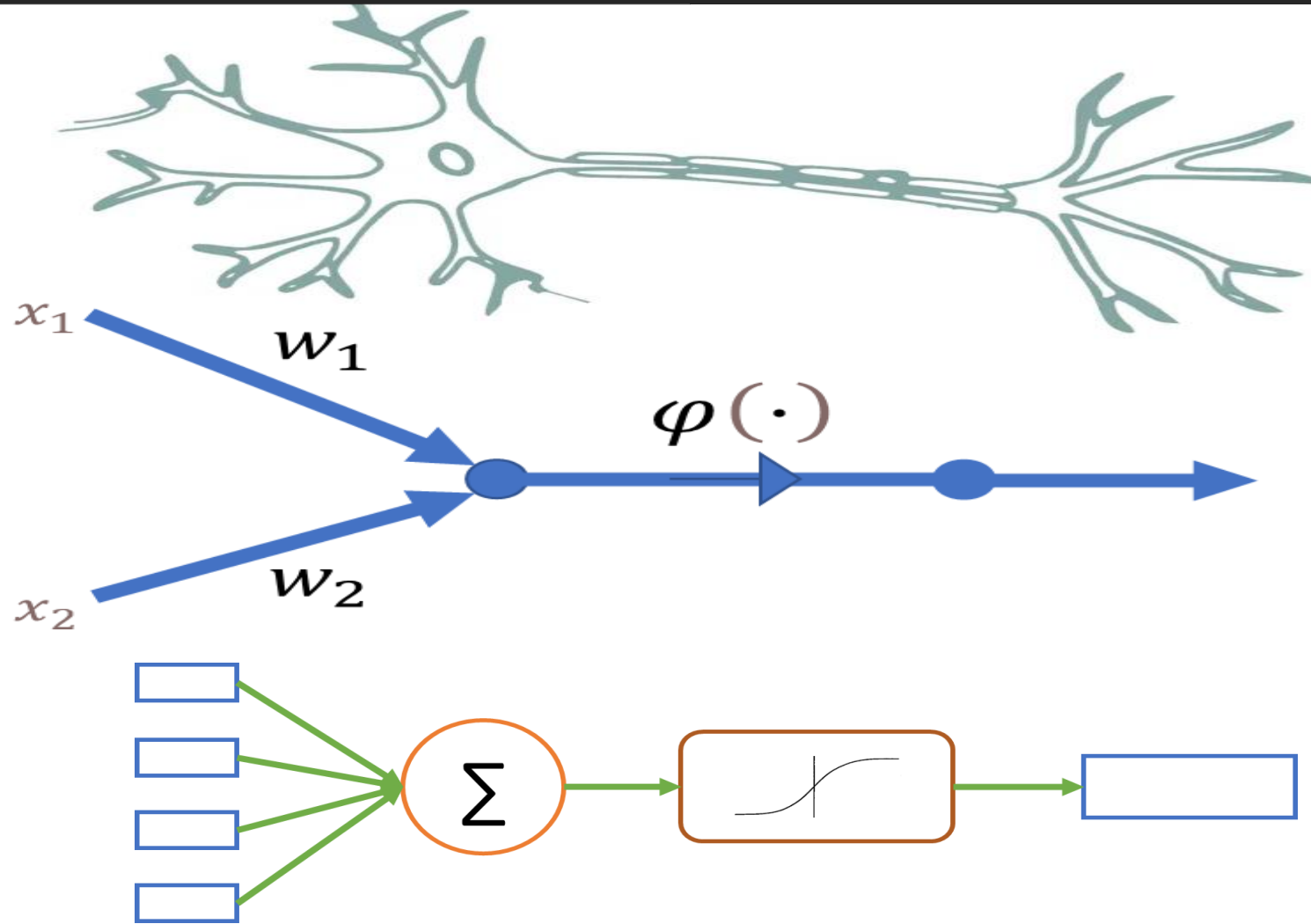








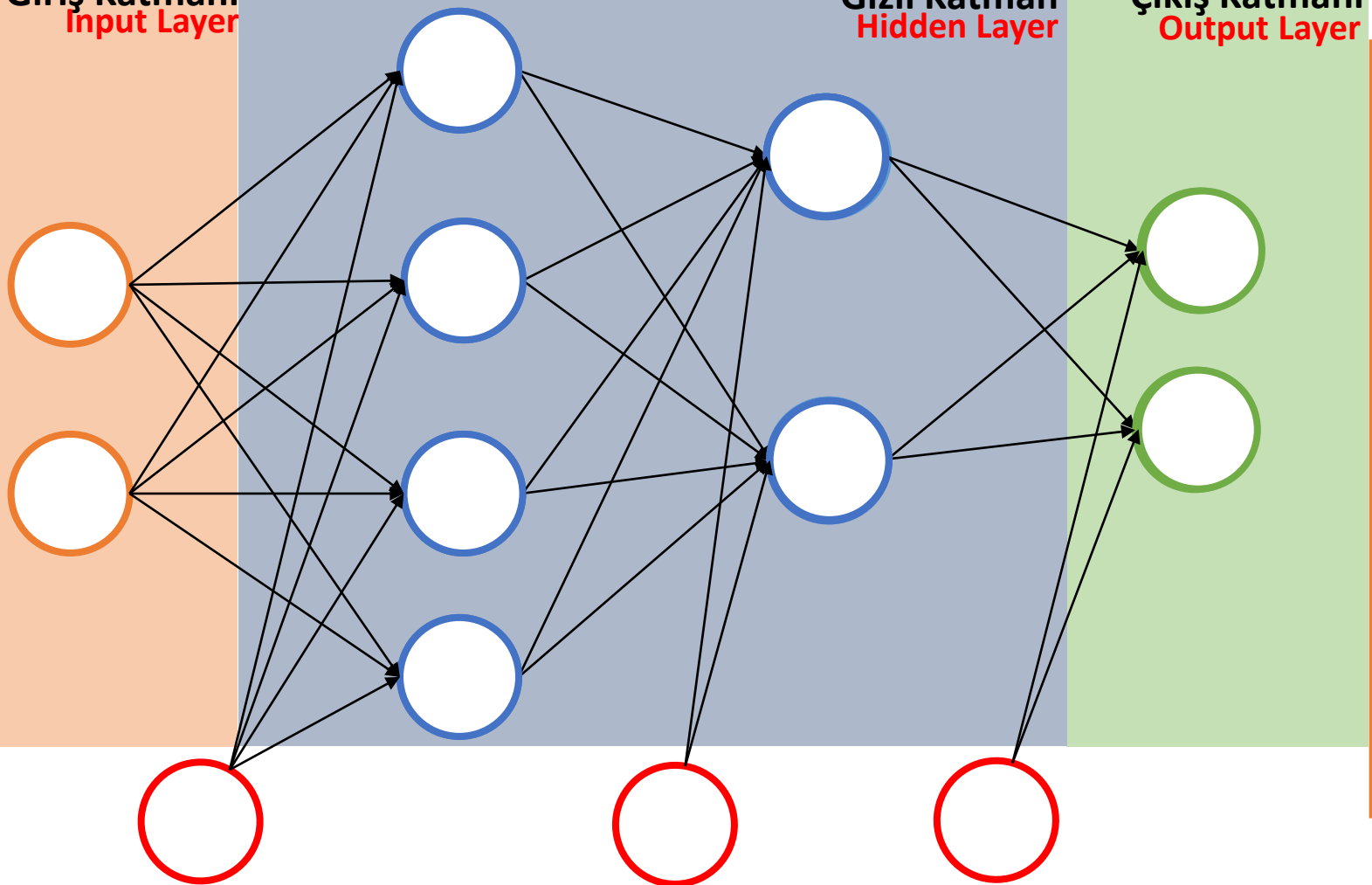


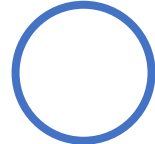
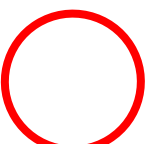


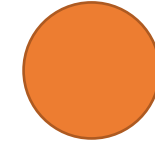
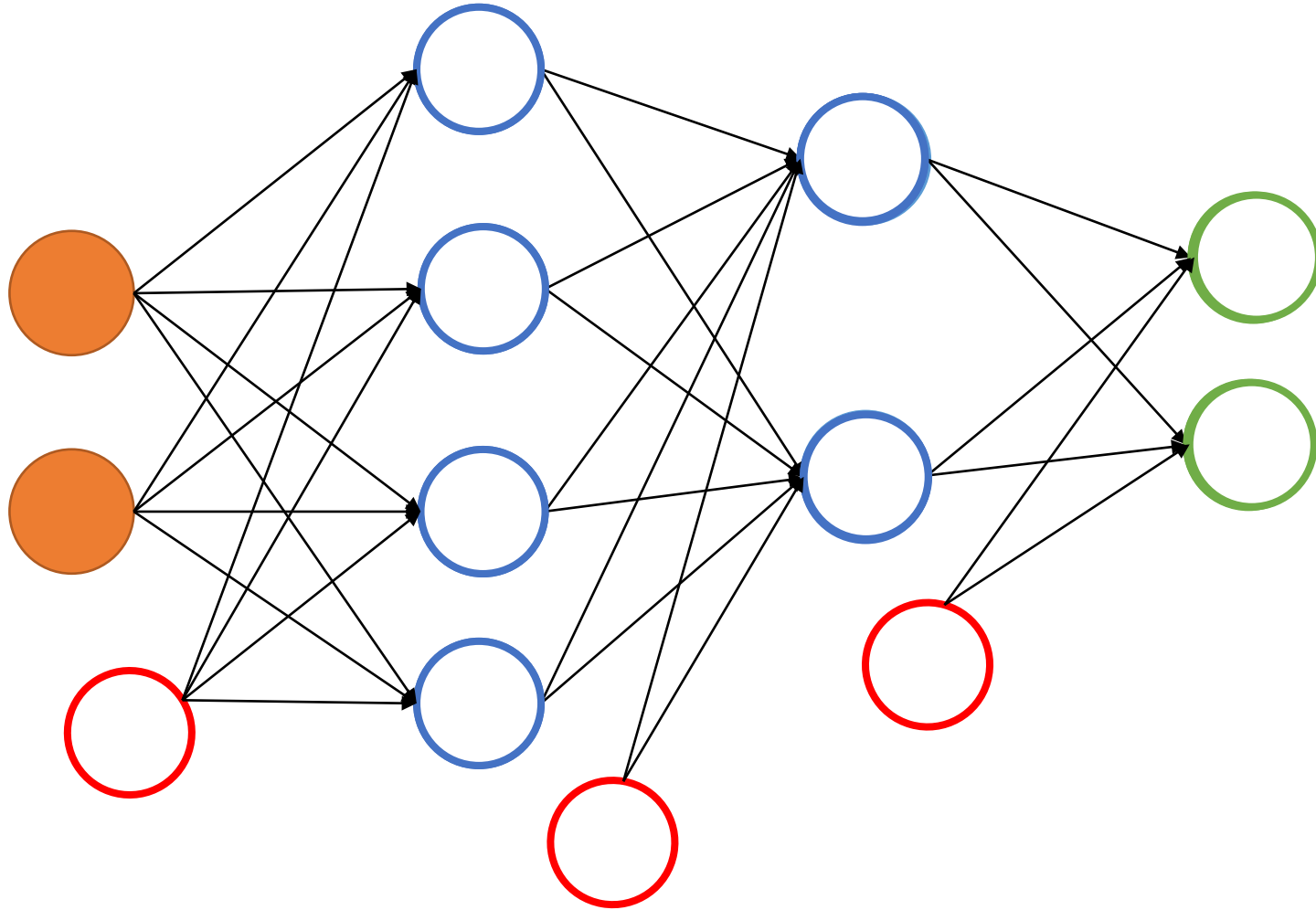
Giriş Katmanı
Input Layer

Gizli Katman
Hidden Layer

Çıkış Katmanı
Output Layer

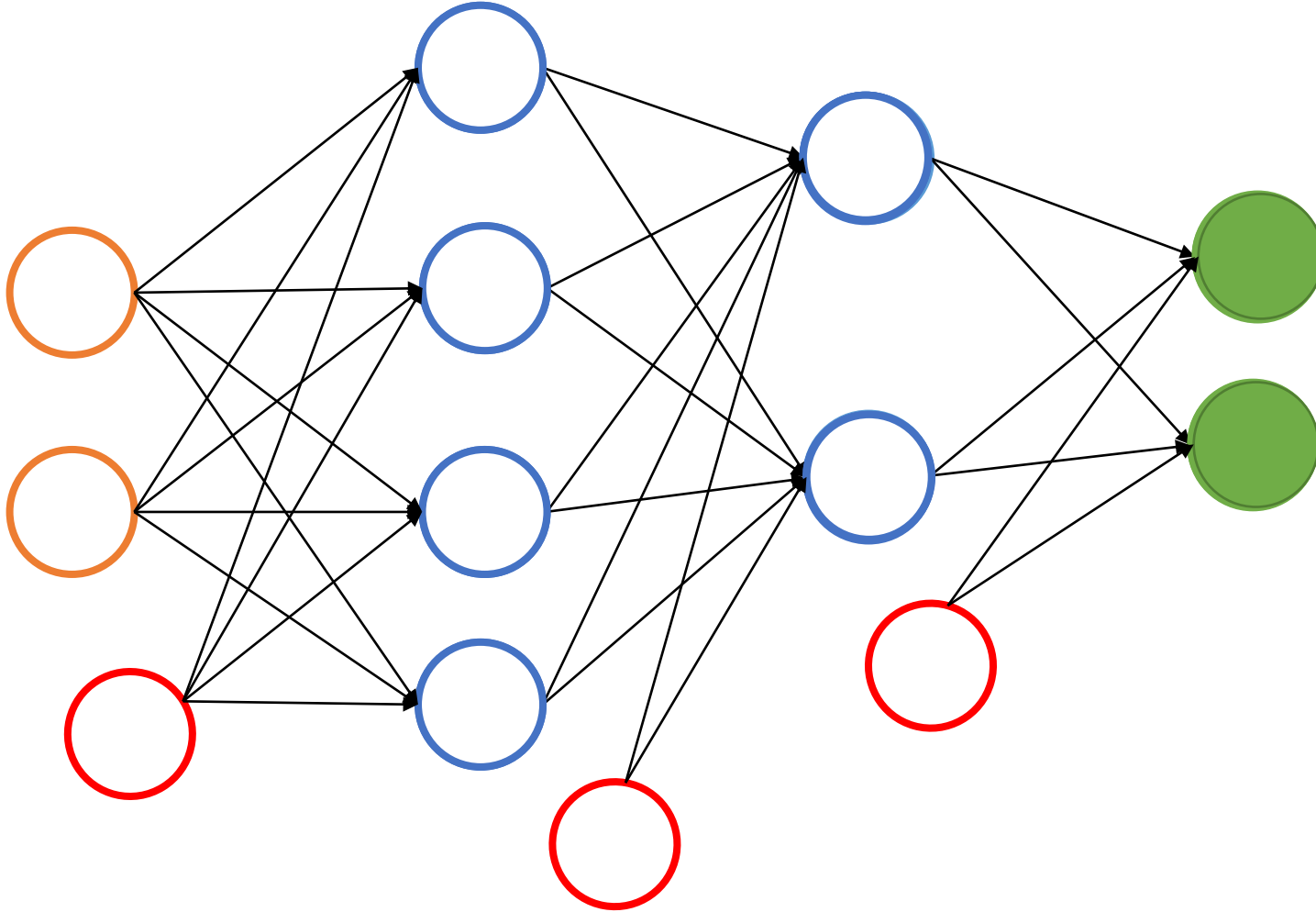


-  **Giriş nöronları**
Input neurons
-  **Çıkış nöronları**
Output neurons
-  **Gizli nöronlar**
Hidden neurons
-  **Bias nöronları**
Bias neurons



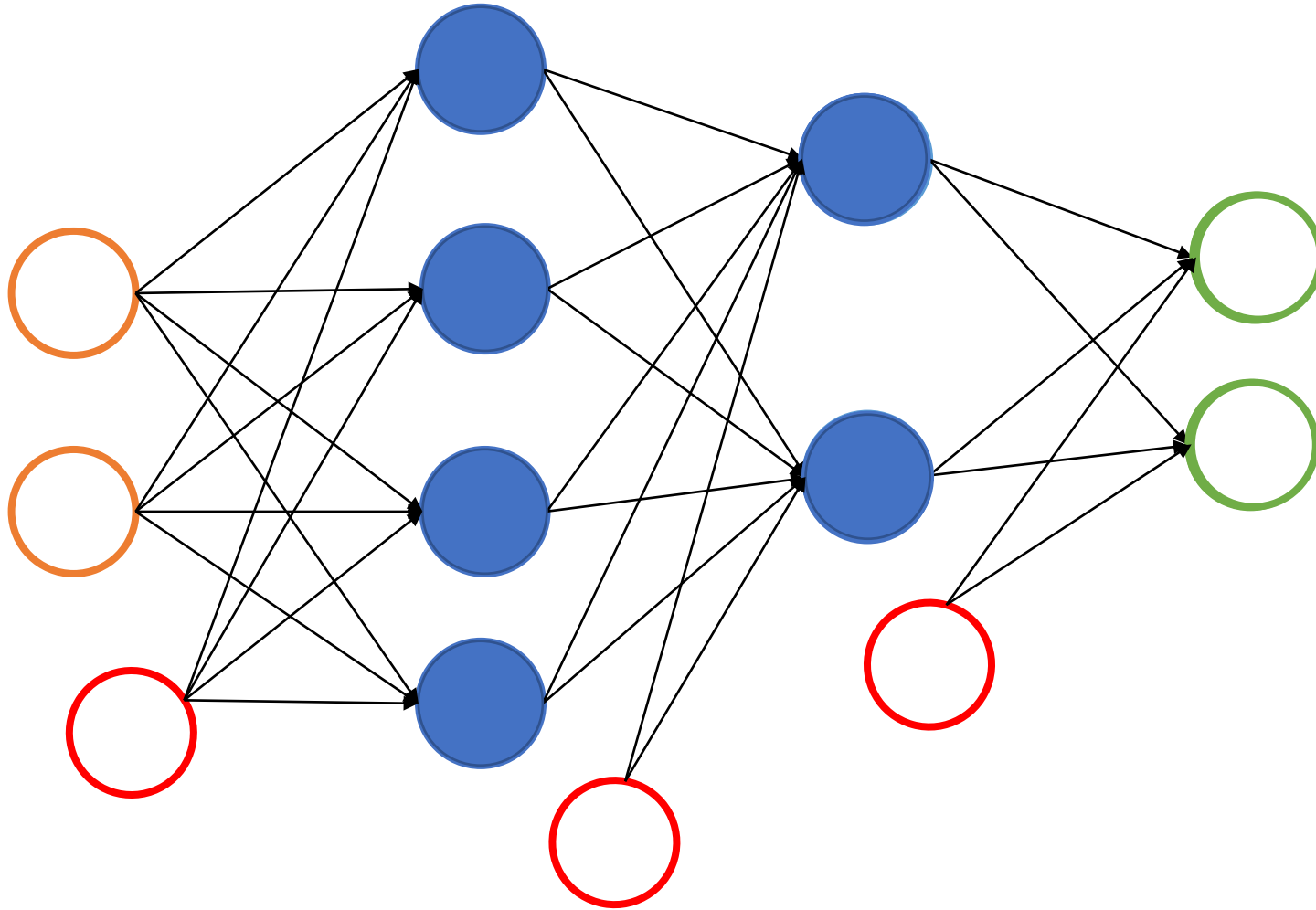
Giriş nöronları
Input neurons

Ağa başlangıç bilgisinin
gönderildiği nöronlardır.



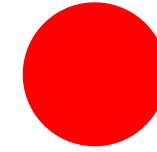
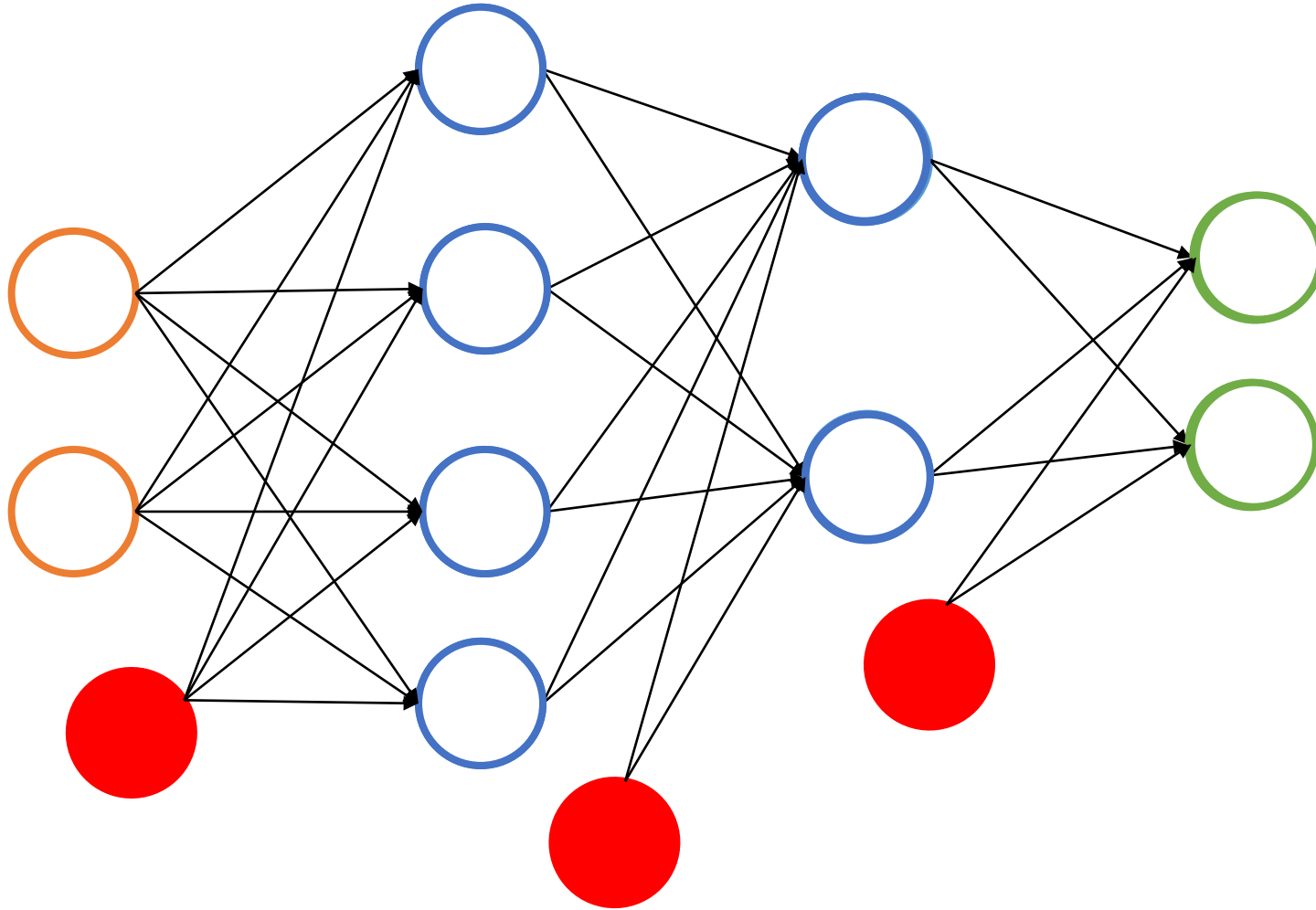
Çıkış nöronları
Output neurons

Ağın sonunda elde edilen çıkış
bilgisini barındıran
nöronlardır.



Gizli nöronlar
Hidden neurons

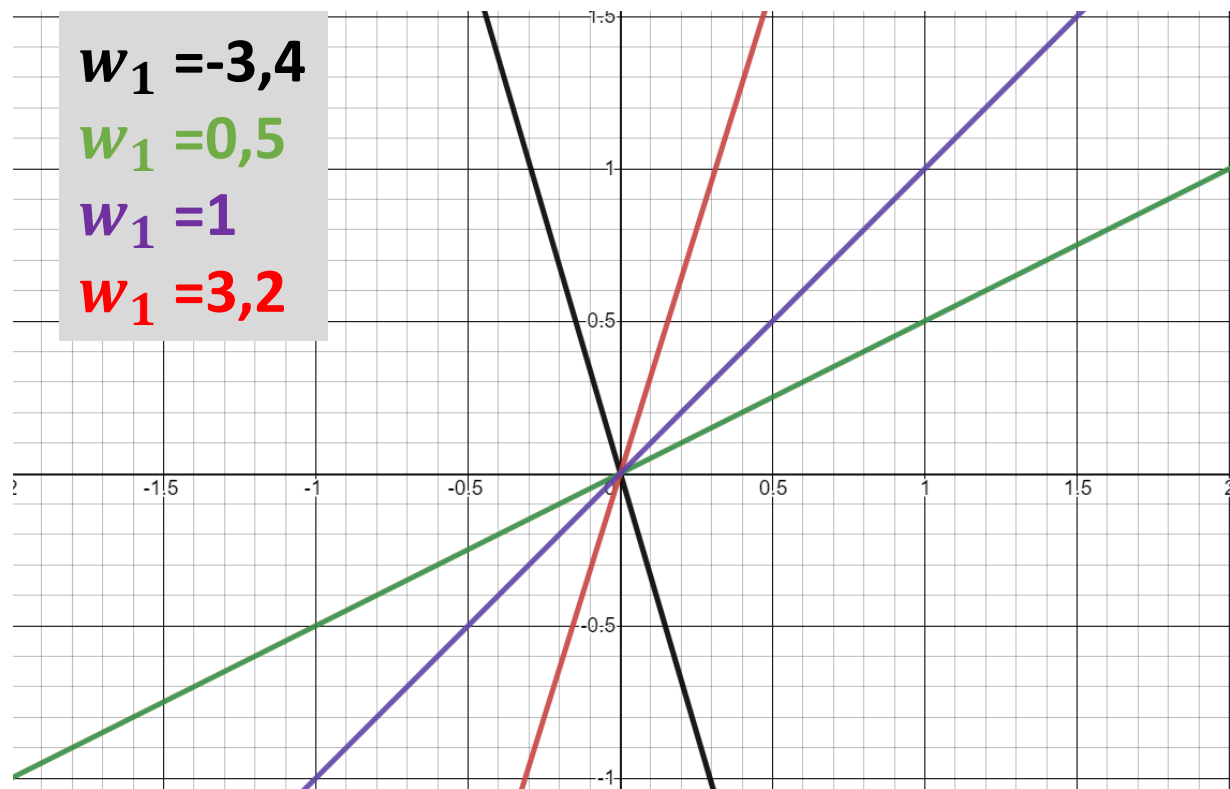
- Giriş, Bias veya kendisinden önceki nöronlarla bağlıdır.
- Çıkışa veya kendinden sonraki nöronlara bağlıdır.
- Giriş ve çıkış arasındaki ilişkiyi belirlemek için kullanılırlar.



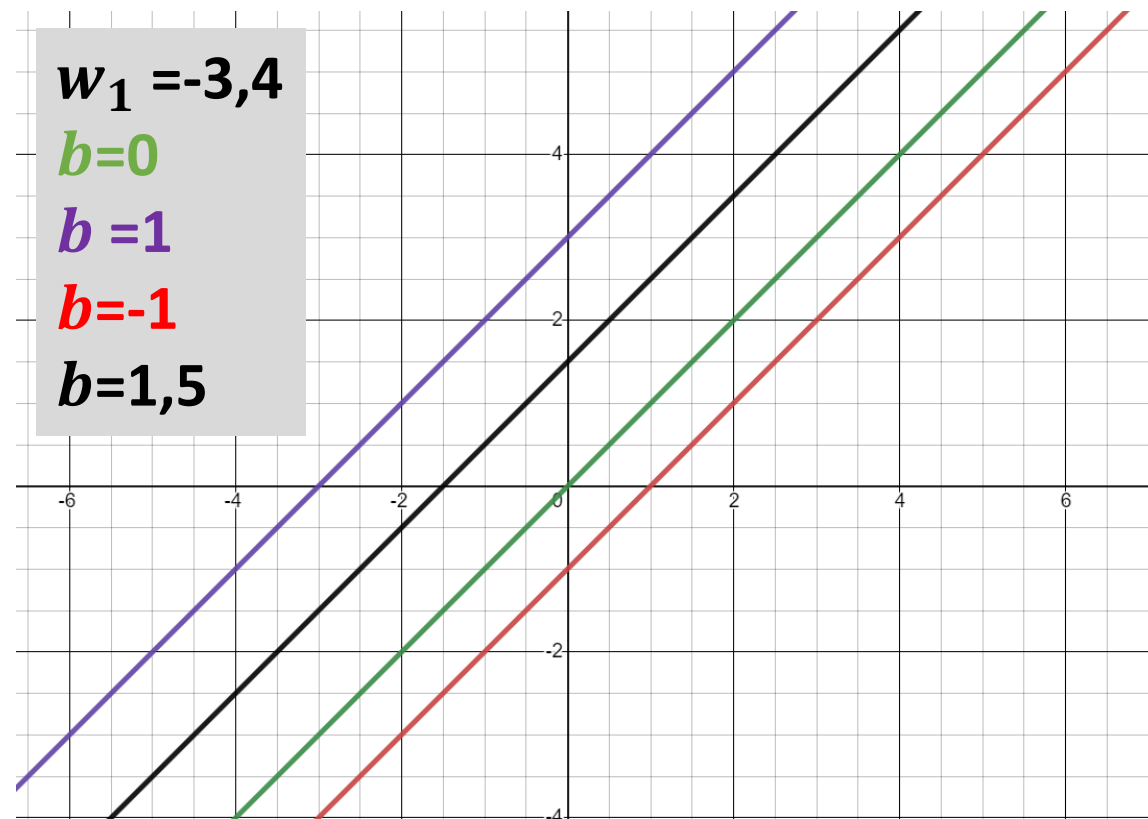
Bias nöronları
Bias neurons

- Nöron çıkıtısını optimize etmek için kullanılan tasarımcı tarafından eklenenebilen nöronlardır.

$$y = x_1 w_1$$

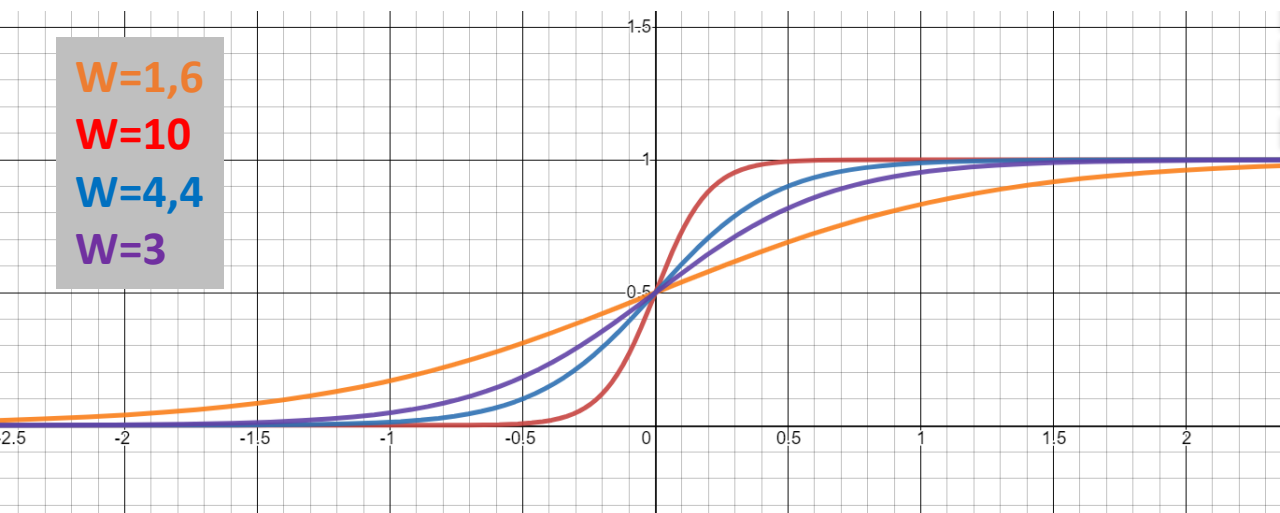


$$y = x_1 w_1 + b$$

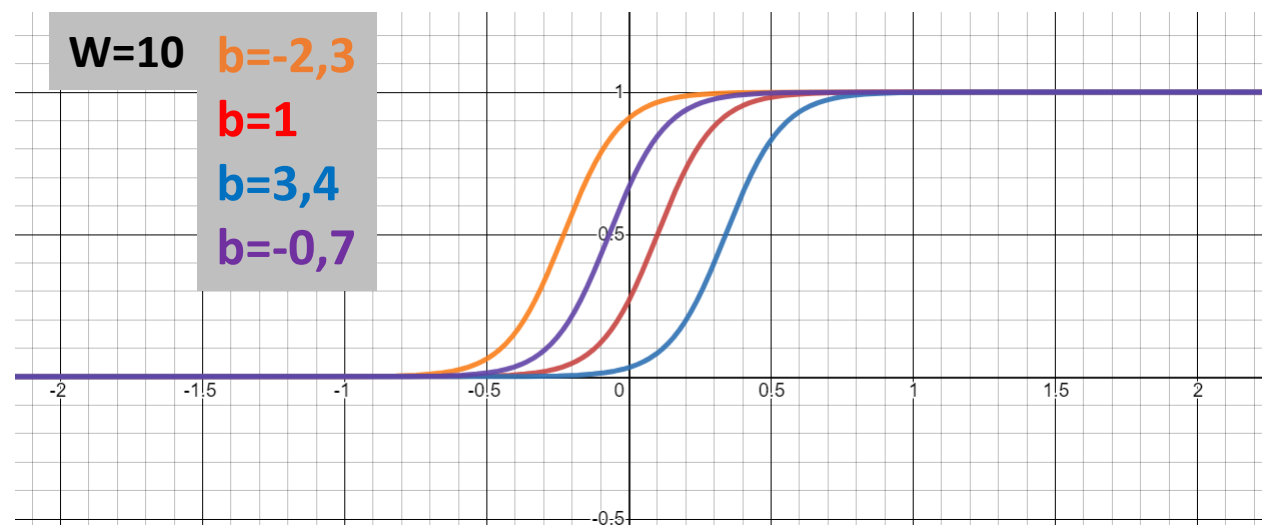


bias
nöronu? | **yapay**
nöron?

$$y = \frac{1}{1 + e^{-xw}}$$

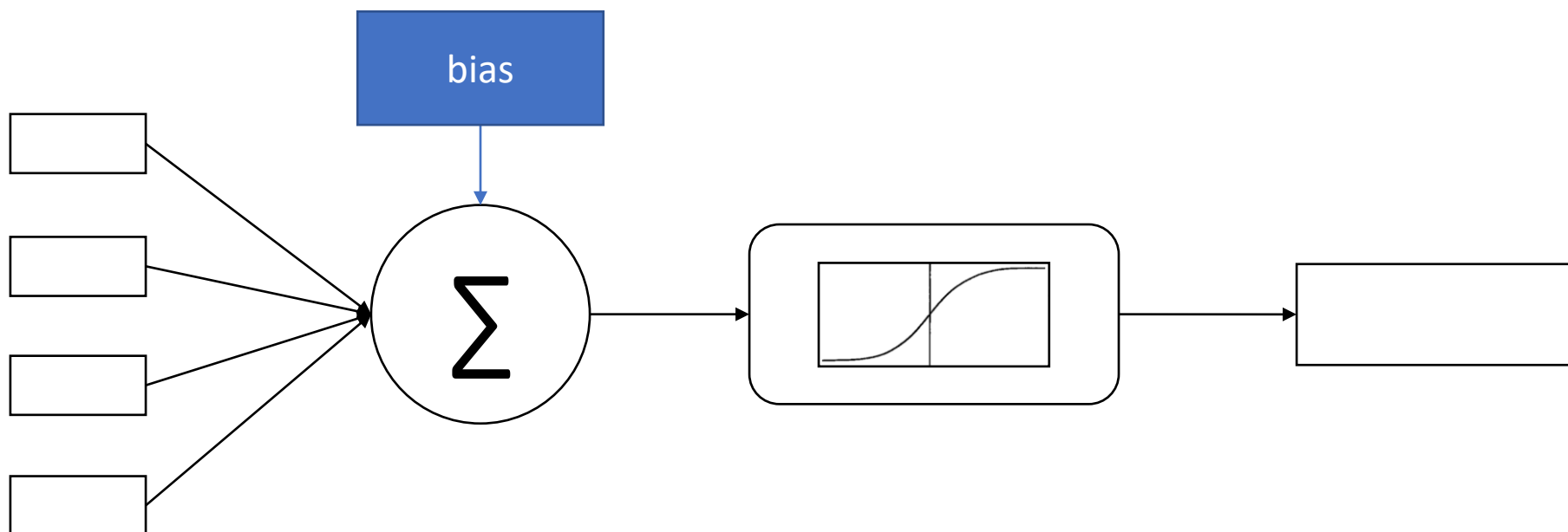


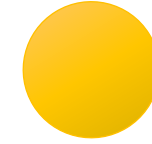
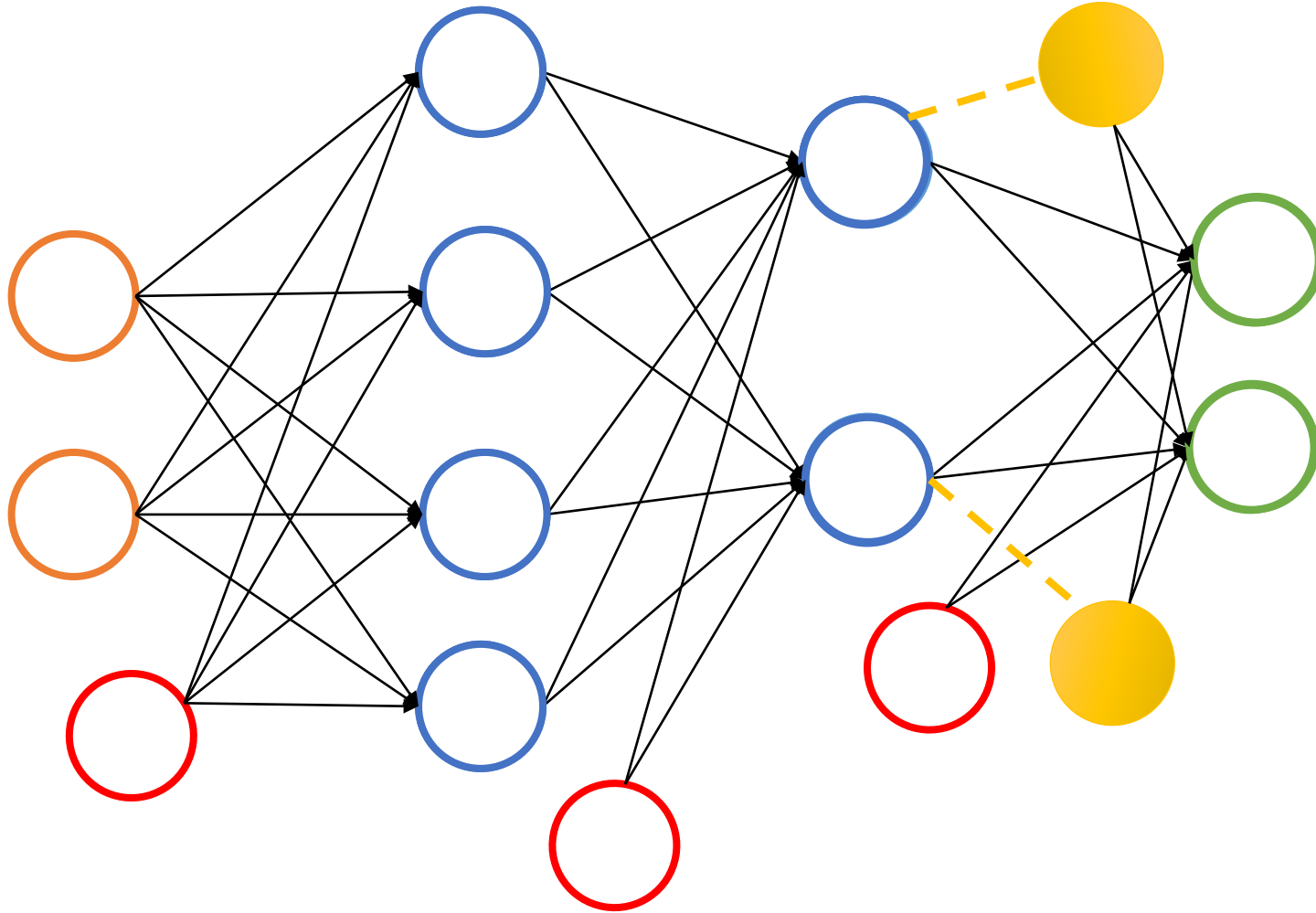
$$y = \frac{1}{1 + e^{-xw+b}}$$



bias
nöronu? | **yapay**
nöron?

Bias eklenmiş nöron yapısı





Konteks nöronları Context neurons

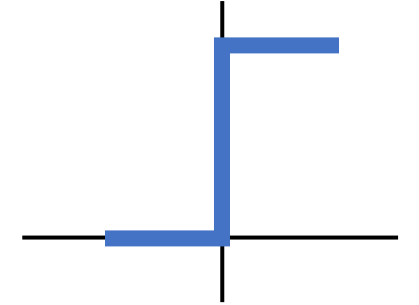
- Sistemin bir önceki durumunu hafızada tutmak için kullanılan kopya nöronlardır.
- Recurrent sinir ağlarında kullanılırlar.
- Video, ses, metin işleme gibi bir önceki zamandaki bilgiye ihtiyaç duyulan uygulamalarda gerekir.



aktivasyon **f**onksiyonu
activation **f**unction

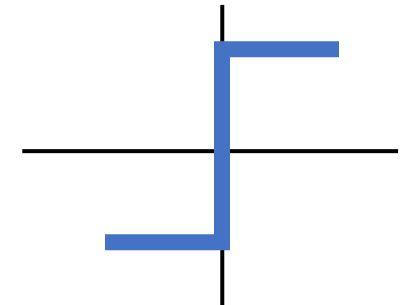
Kesin sınır
hardlimit
Step function

$$\begin{cases} a = 0, & n < 0 \\ a = 1, & n \geq 0 \end{cases}$$



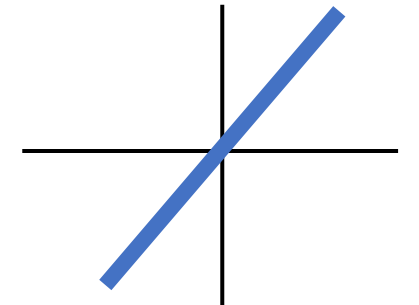
Simetrik kesin sınır
Symetric hardlimit

$$\begin{cases} a = -1, & n < 0 \\ a = 1, & n \geq 0 \end{cases}$$



Doğrusal
linear

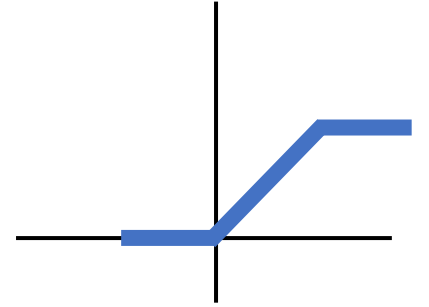
$$a = n$$



aktivasyon
fonksiyonu?

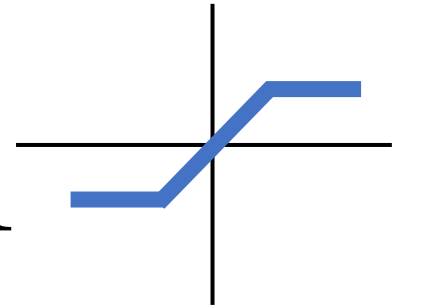
Doygun Doğrusal
saturated linear

$$\begin{cases} a = 0, & n < 0 \\ a = n, & 1 \geq n \geq 0 \\ a = 1, & n > 1 \end{cases}$$



Simetrik
Doygun Doğrusal
symetric
saturated linear

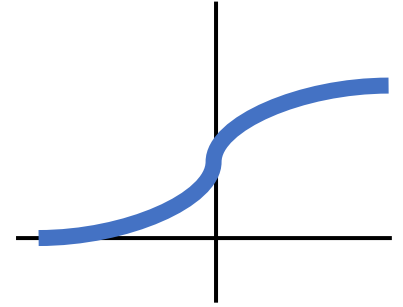
$$\begin{cases} a = -1, & n < -1 \\ a = n, & 1 \geq n \geq -1 \\ a = 1, & n > 1 \end{cases}$$



aktivasyon
fonksiyonu?

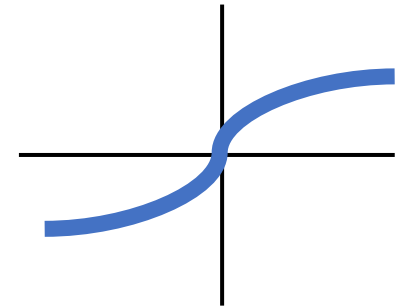
Log Sigmoid
log sigmoid

$$a = \frac{1}{1 + e^{-n}}$$



**Hiberbolik
Tanjant Sigmoid**
hyperbolic
tangent sigmoid

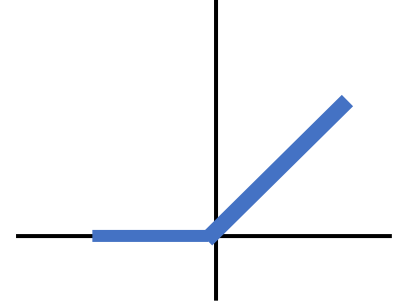
$$a = \frac{e^n - e^{-n}}{e^n + e^{-n}}$$



aktivasyon
fonksiyonu?

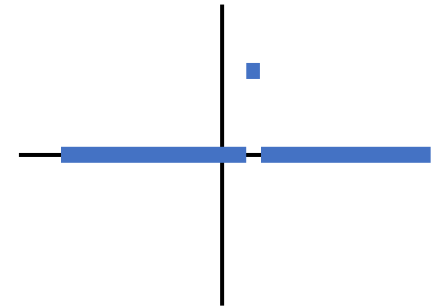
Pozitif doğrusal
positive linear
ReLU

$$\begin{cases} a = 0, & n < 0 \\ a = n, & n \geq 0 \end{cases}$$



Rekabetçi
competitive

$$\begin{cases} a = 0, & \text{diğer tüm nöronlar} \\ a = 1, & \text{en büyük } n \text{ değerli nöron} \end{cases}$$



aktivasyon
fonksiyonu?

Softmax softmax

$$\varphi_i = \frac{e^{z_i}}{\sum_{j \in group} e^{z_j}}$$

Sınıflandırma ağlarında kullanılır.

Genelde çıkış katmanında bulunur.

Sınıfın üyeliklerine o grubun % kaç ihtimalle olduğu bilgisini gönderir.

$$\varphi_i = \frac{e^{z_i}}{\sum_{j \in group} e^{z_j}}$$

```
def softmax(giris):  
    toplam=0  
    for j in giris:  
        toplam=toplam+np.exp(j)  
  
    olasilik=[]  
    for i in range (len(giris)):  
        olasilik.append(np.exp(giris[i])/toplam)  
    return olasilik
```

```
def softmax2(X):  
    expo = np.exp(X)  
    expo_sum = np.sum(np.exp(X))  
    return expo / expo_sum
```

Softmax python'da yazılmış fonksiyon

Test kodu ve ekran çıktısı

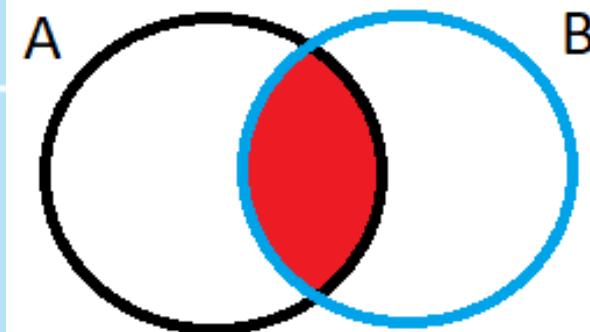
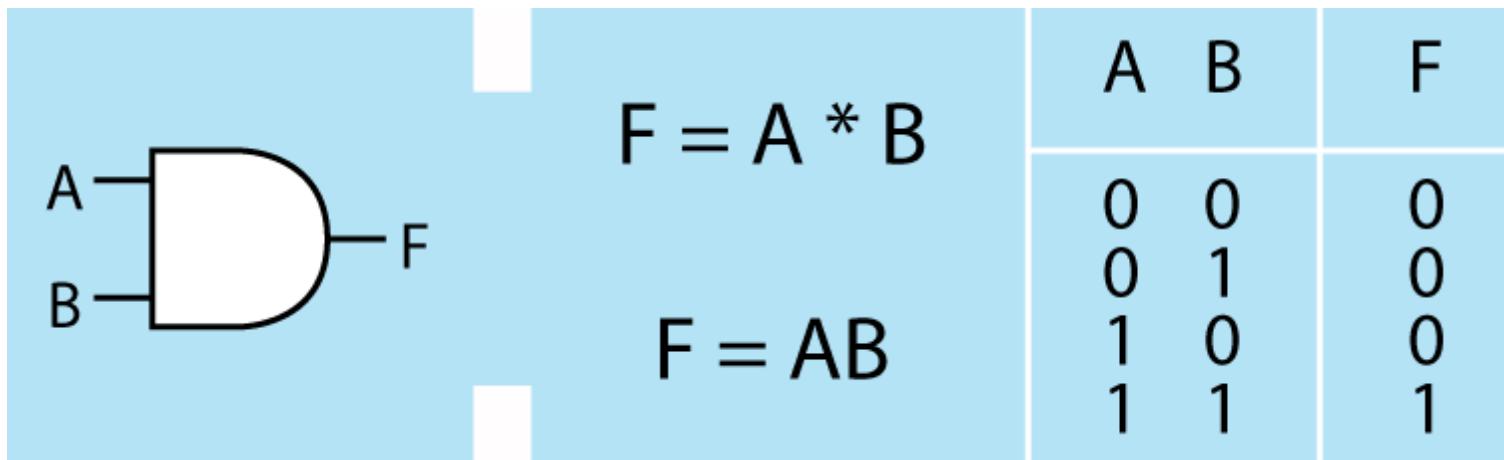
```
vector=np.array([0.5,0.32,0.9,0.1])  
  
print(softmax(vector))  
k = cv.waitKey(0)  
if k == 27:  
    cv.destroyAllWindows()
```

```
C:\Users\*****\anaconda3\envs\openCV\pyth  
on.exe  
C:/Users/*****/PycharmProjects/goru/ann.p  
y  
[0.25016166978790433,  
0.20895259081073966,  
0.37319735739277565,  
0.1676883820085804]
```



xor kapısı xor gate

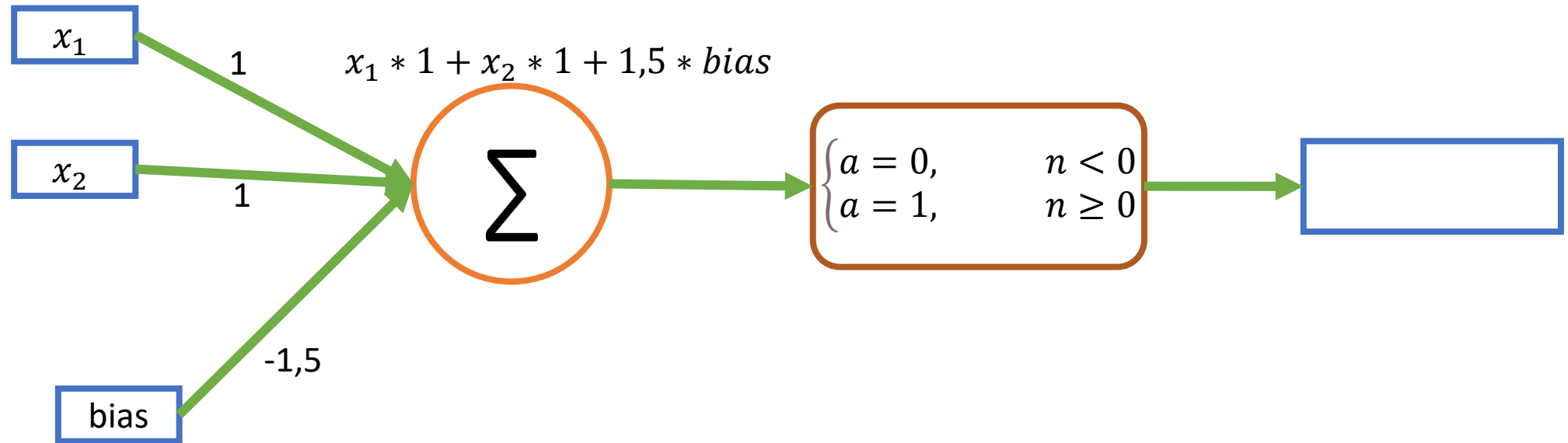
AND/VE kapısı



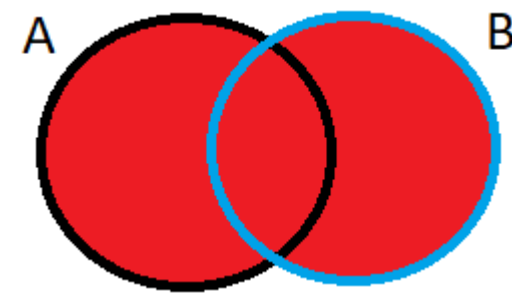
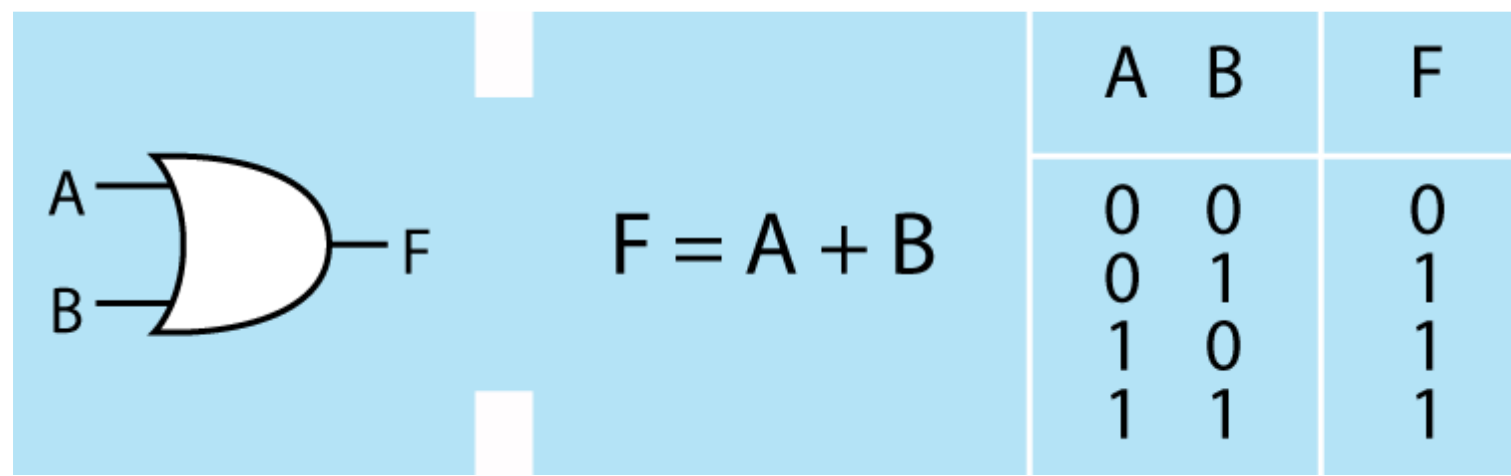
girişler			ağırlıklar				çıkış	
b	x1	x2	wb	w1	w2	toplam	aktivasyon	olması gereken
1	0	0	-1,5	1	1	-1,5	0	0
1	0	1				-0,5	0	0
1	1	0				-0,5	0	0
1	1	1				0,5	1	1

xor kapısı !
örneği!

AND/VE kapısı



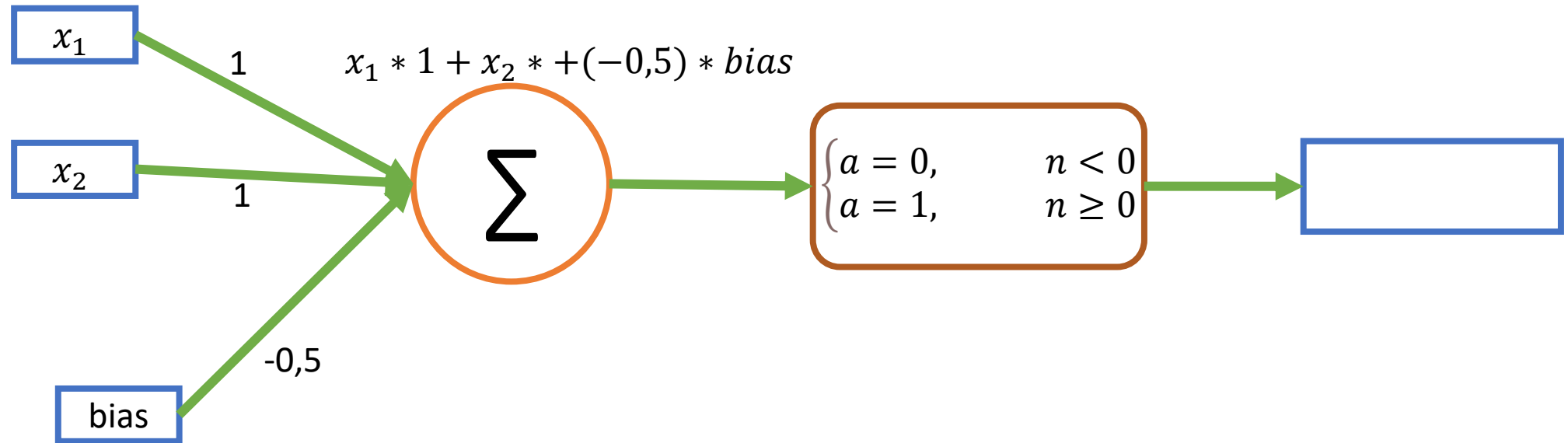
OR/VEYA kapısı



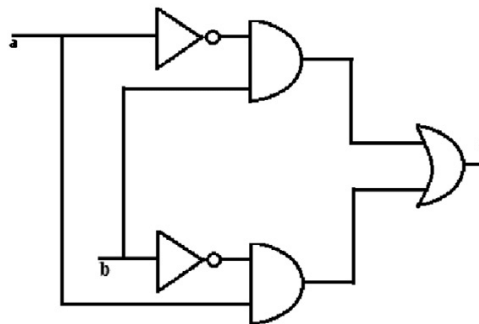
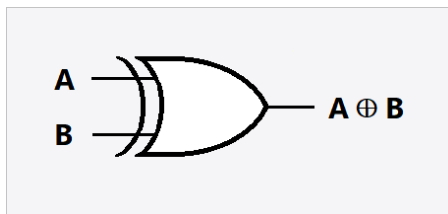
girişler			ağırlıklar				çıkış	
b	x1	x2	wb	w1	w2	toplam	aktivasyon	olması gereken
1	0	0	-0,5	1	1	-0,5	0	0
1	0	1				0,5	1	1
1	1	0				0,5	1	1
1	1	1				1,5	1	1

xor kapısı !
örneği!

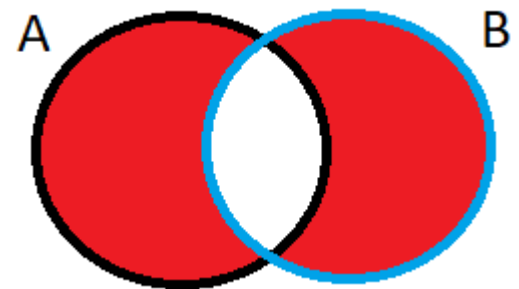
OR/VEYA kapısı



XOR/ ÖZEL VEYA kapısı



xor		
a	b	f
0	0	0
0	1	1
1	0	1
1	1	0

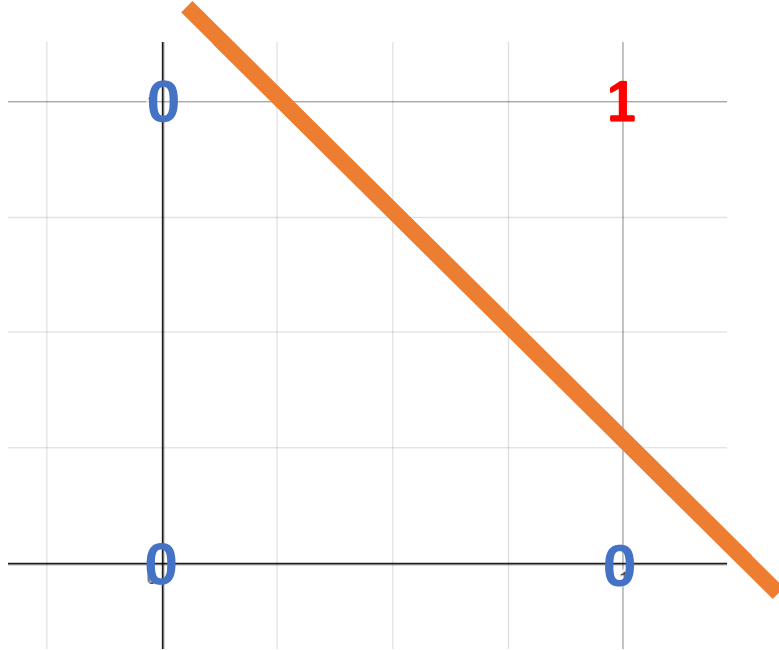


$$f(A, B) = \bar{A}B + \bar{B}A$$

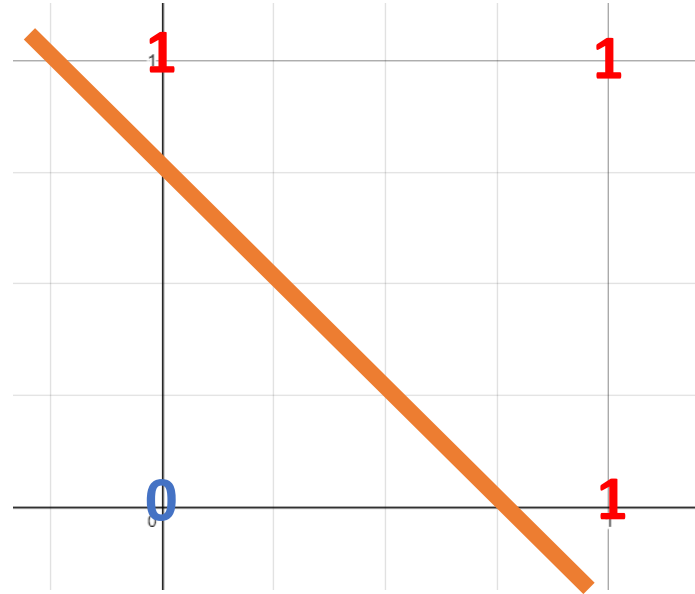
girişler			ağırlıklar				çıkış	
b	x1	x2	wb	w1	w2	toplam	aktivasyon	olması gereken
1	0	0	-0,75	2	1	-0,75	0	0
1	0	1				0,25	1	1
1	1	0				1,25	1	1
1	1	1				2,25	1	0

xor kapısı !
örneği!

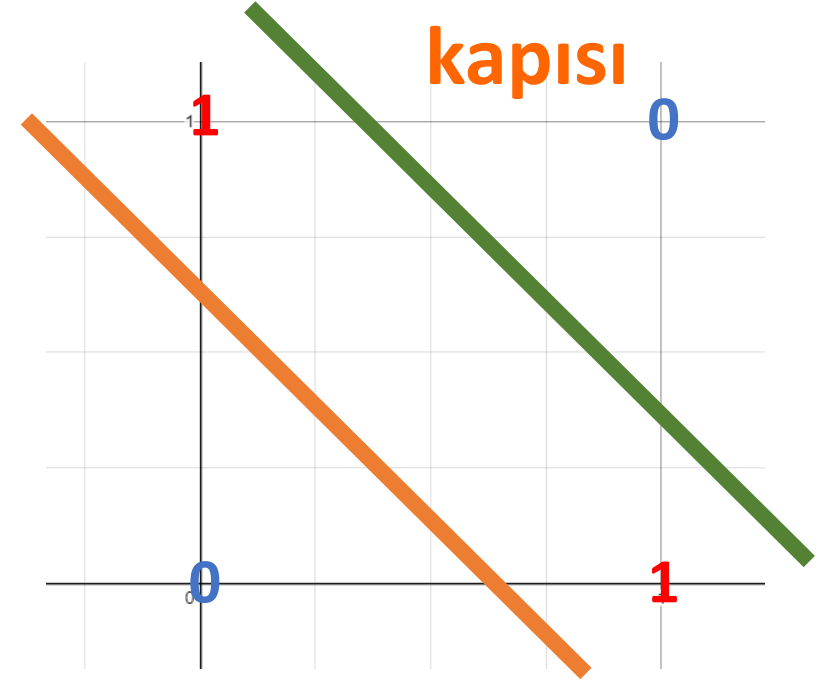
AND/VE kapısı



OR/VEYA kapısı



XOR/ ÖZEL VEYA kapısı



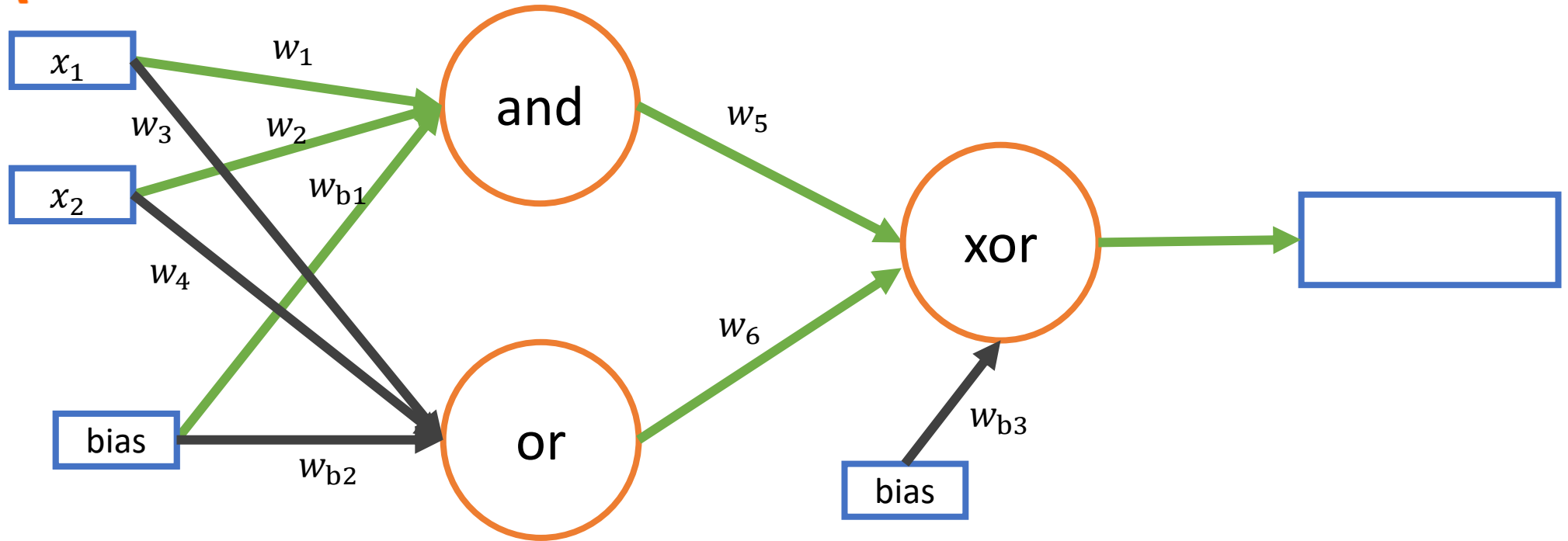
xor kapısı !
örneği !

XOR/ ÖZEL VEYA kapısı

a	b	ab	a+b	aXORb
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

xor kapısı !
örneği!

XOR/ ÖZEL VEYA kapısı



xor kapısı !
örneği!

XOR/ÖZEL VEYA kapısı

AND VE KAPISI

girişler			ağırlıklar				çıkış	
b	x1	x2	wb1	w1	w2	toplam	aktivasyon	olması
1	0	0	-1,5	1	1	-1,5	0	0
1	0	1				-0,5	0	0
1	1	0				-0,5	0	0
1	1	1				0,5	1	1

OR

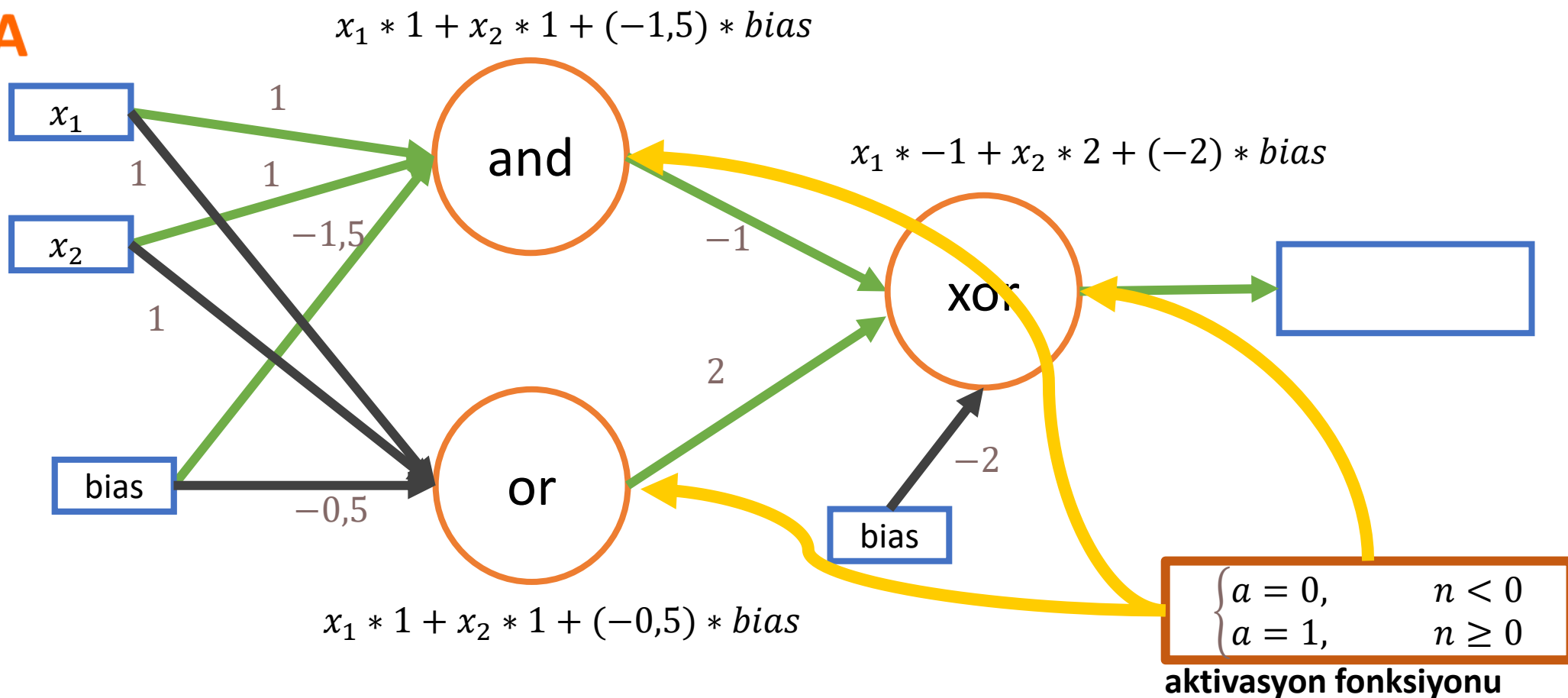
ağırlıklar				çıkış	
wb2	w3	w4	toplam	aktivasyon	olması
-0,5	1	1	-0,5	0	0
			0,5	1	1
			0,5	1	1
			1,5	1	1

XOR

b	and çıkış	or çıkış	wb3	x5	x6	toplam	aktivasyon	olması gereken
1	0	0	-2	-1	2	-2	0	0
1	0	1				0	1	1
1	0	1				0	1	1
1	1	1				-1	0	0

xor kapısı !
örneği!

XOR/ ÖZEL VEYA kapısı



xor kapısı!
örneği!

Ağırlıkların elle hesaplanması zor bir iş...
Peki bilgisayarlar bu ağırlıkları nasıl hesaplıyor?



Ysa nasıl Öğrenir?
How ann learns?



hataerror

$$\text{Hata} = \text{ölçülen değer} - \text{olması gereken değer}$$

Error = actual value - ideal value

Yerel Hata

local error

- Her tek bir nöronun mevcut durumu ve olması gereken değeri arasındaki farktır.
- Hata fonksiyonu ile hesaplanır.

Genel Hata

global error

- Yerel hataların bir araya gelmesi ile oluşan ağın toplam hatasıdır.
- Sinir Ağının eğitim setine ne kadar iyi cevap verdiğinin göstergesidir.

Hata kareler toplamı

Ortalama Kareler Hatası

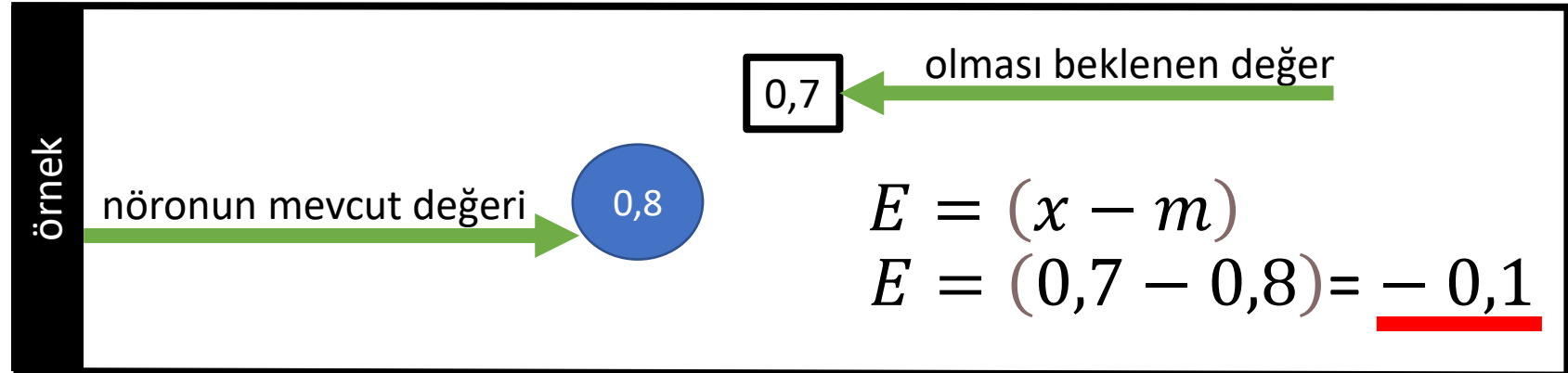
Ortalama Karekök Hatası

Yerel Hata local error

Doğrusal Hata Fonksiyonu Linear Error Function

$$E = (x - m)$$

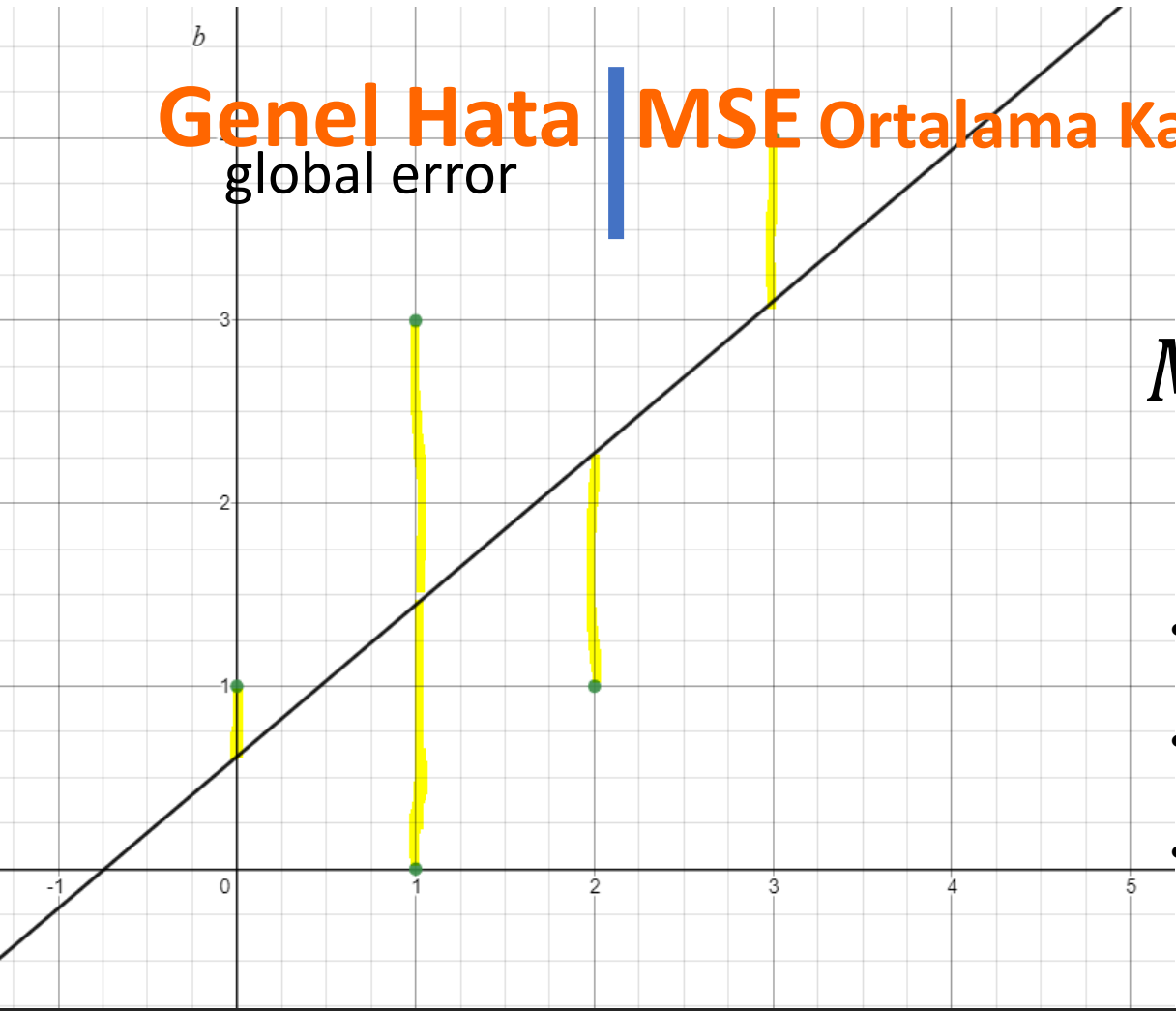
x : olması beklenen değer
 m : nöronun mevcut değeri



$|E|$

Hatanın mutlak değerinin büyüklüğü arzu edilen değerden ne kadar uzaklaşıldığını gösterir.
Hatanın işareti ise ağırlık düzeltmelerinde artış veya azalış yapılması gerektiği hakkında bilgi verir.

$\pm E$



$$MSE = \frac{1}{n} \sum_{i=1}^n E^2$$

E : yerel hata

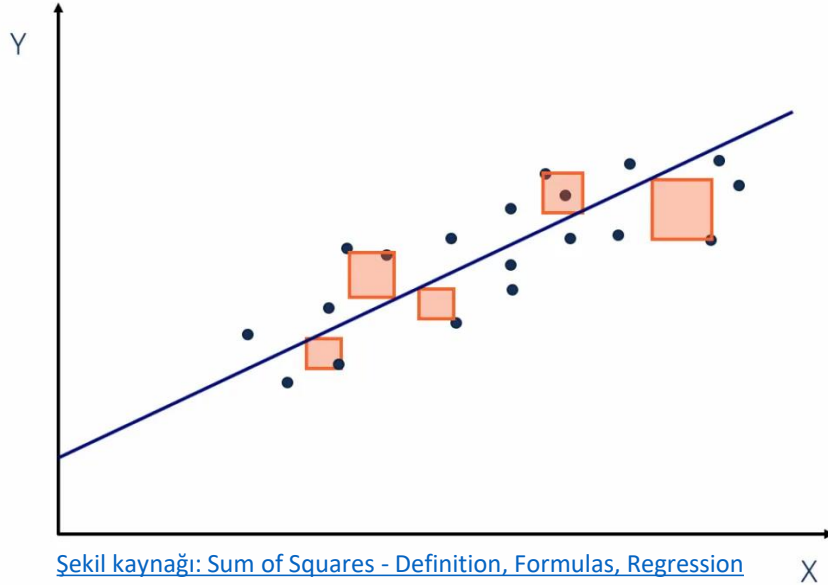
n : yerel hata sayısı

- MSE regresyon çizgisinin veri setindeki noktalara ne kadar uzakta olduğunu söyler.
- Regresyon çizgisi üzerindeki noktaların veri setindeki noktalara uzaklığı hesaplanarak hata bulunur.
- Kare kullanılması hem negatif işareti kaldırır hem de değerler arasındaki farkı arttıran bir ağırlık görevi görür.

Genel Hata | SSE Hata Kareler Toplamı

global error

Sum of Squares Error
Residual sum of squares



Şekil kaynağı: Sum of Squares - Definition, Formulas, Regression Analysis (corporatefinanceinstitute.com)

$$SSE = \frac{1}{2} \sum_{i=1}^n E^2 \quad E: \text{yerel hata}$$

- Hataların kareleri toplamı hata sayısına değil 2'ye bölünür.
- Genel olarak, daha düşük bir Hata kareler toplamı, regresyon modelinin verileri daha iyi açıklayabildiğini gösterirken, daha yüksek bir hata kareler toplamı, modelin verileri zayıf bir şekilde açıkladığını gösterir.
- Bazı eğitim algoritmalarında kullanılır. Levenburg-Marguardt Algoritması(LMA)

Genel Hata
global error

RMSE Kök Ortalama Kare
Hata

Root Mean Square Error

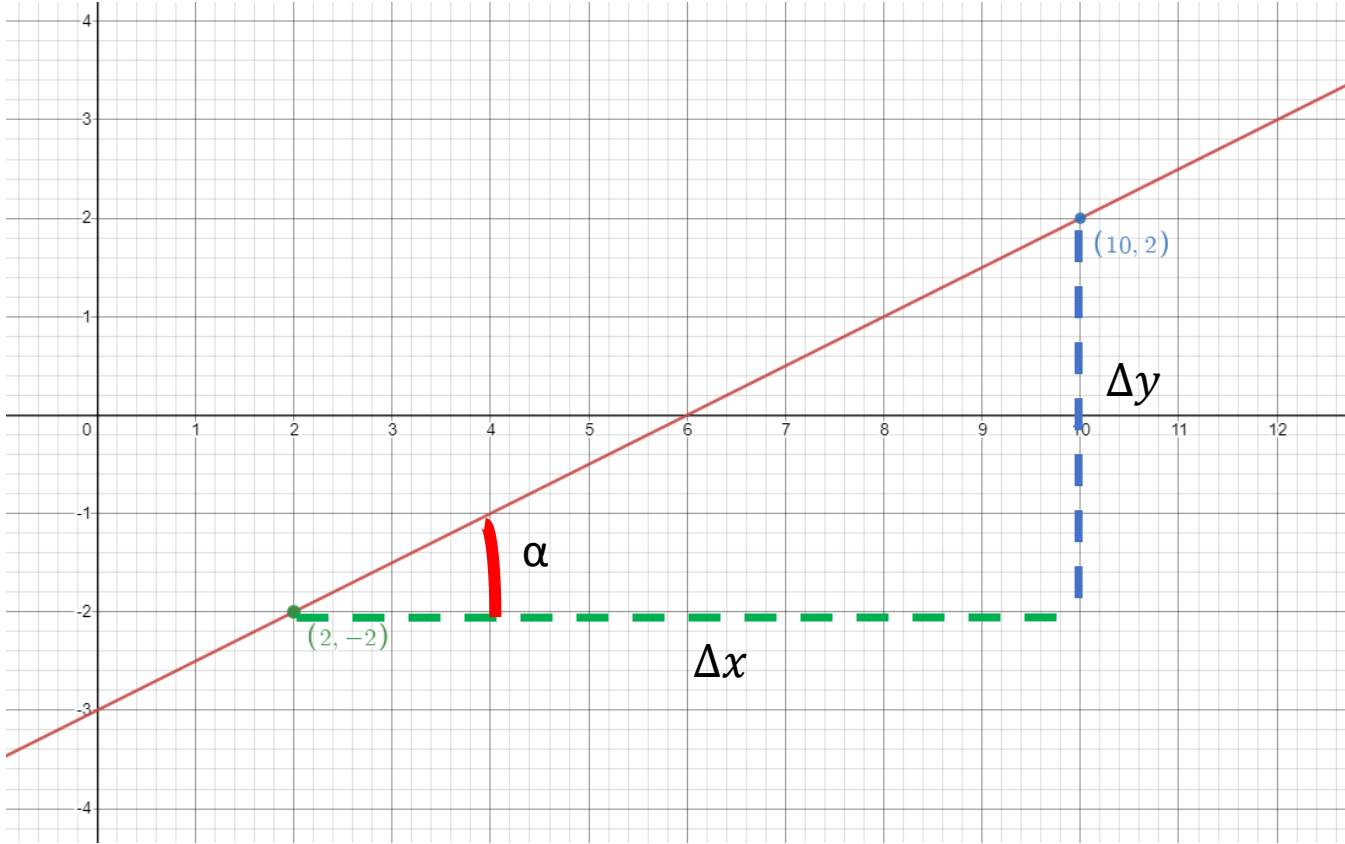
$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n E^2} \quad E: \text{yerel hata}$$

- Elektronik sistemlerde daha çok kullanılır.
- Genel olarak, daha düşük bir Hata kareler toplamı, regresyon modelinin verileri daha iyi açıklayabildiğini gösterirken, daha yüksek bir hata kareler toplamı, modelin verileri zayıf bir şekilde açıkladığını gösterir.
- Bazı eğitim algoritmalarında kullanılır.
Levenburg-Marguardt Algoritması(LMA)

genel hata ? hata ?
golobal error error



eğimslope



$$\text{eğim} = \frac{y' \text{teki değişim}}{x' \text{deki değişim}}$$

$$m = \frac{\Delta y}{\Delta x} = \frac{y_2 - y_1}{x_2 - x_1}$$

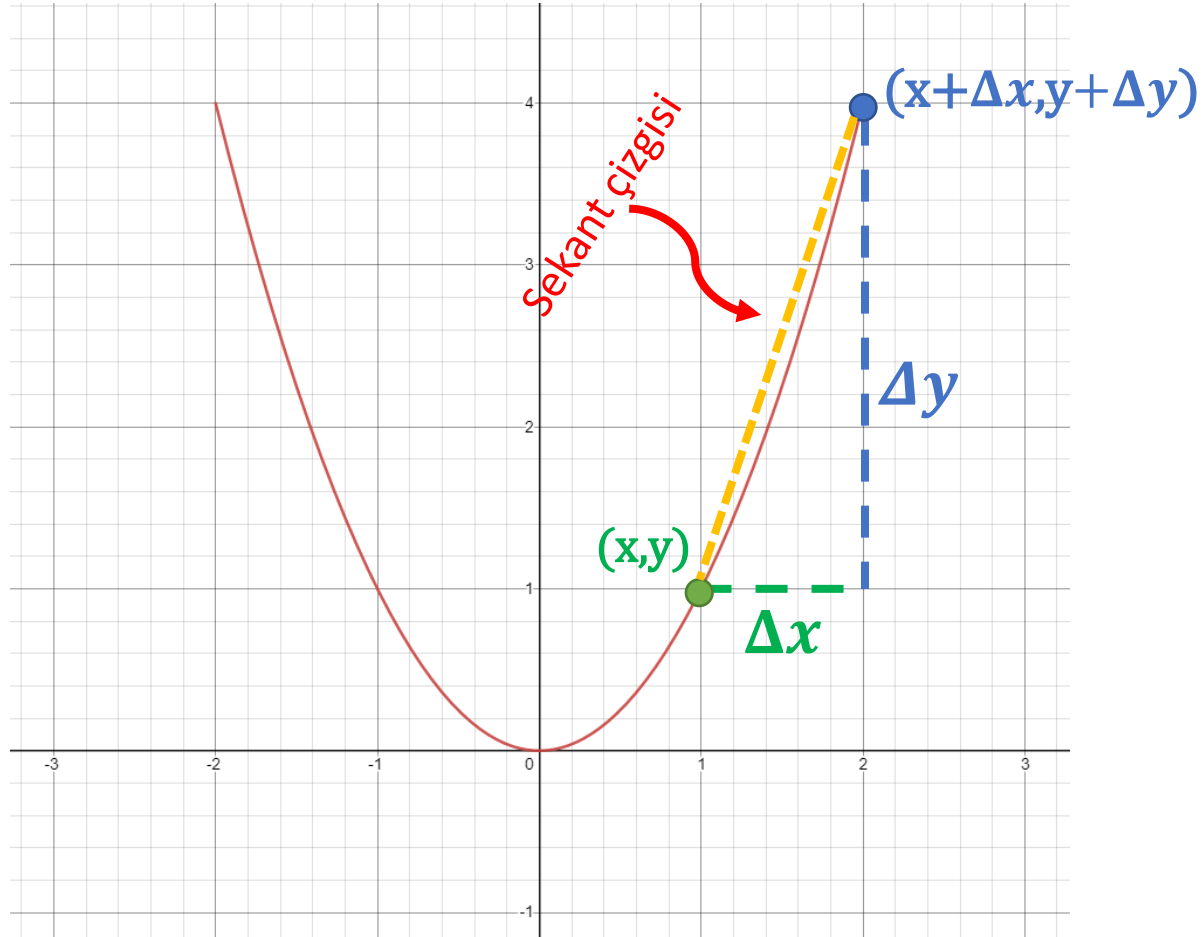
$$m = \tan(\alpha)$$

$$y = mx + b$$

eğim

Y eksenini kesim noktası

doğrunun eğimi ?
Slope of a line ?
eğim ?
slope ?



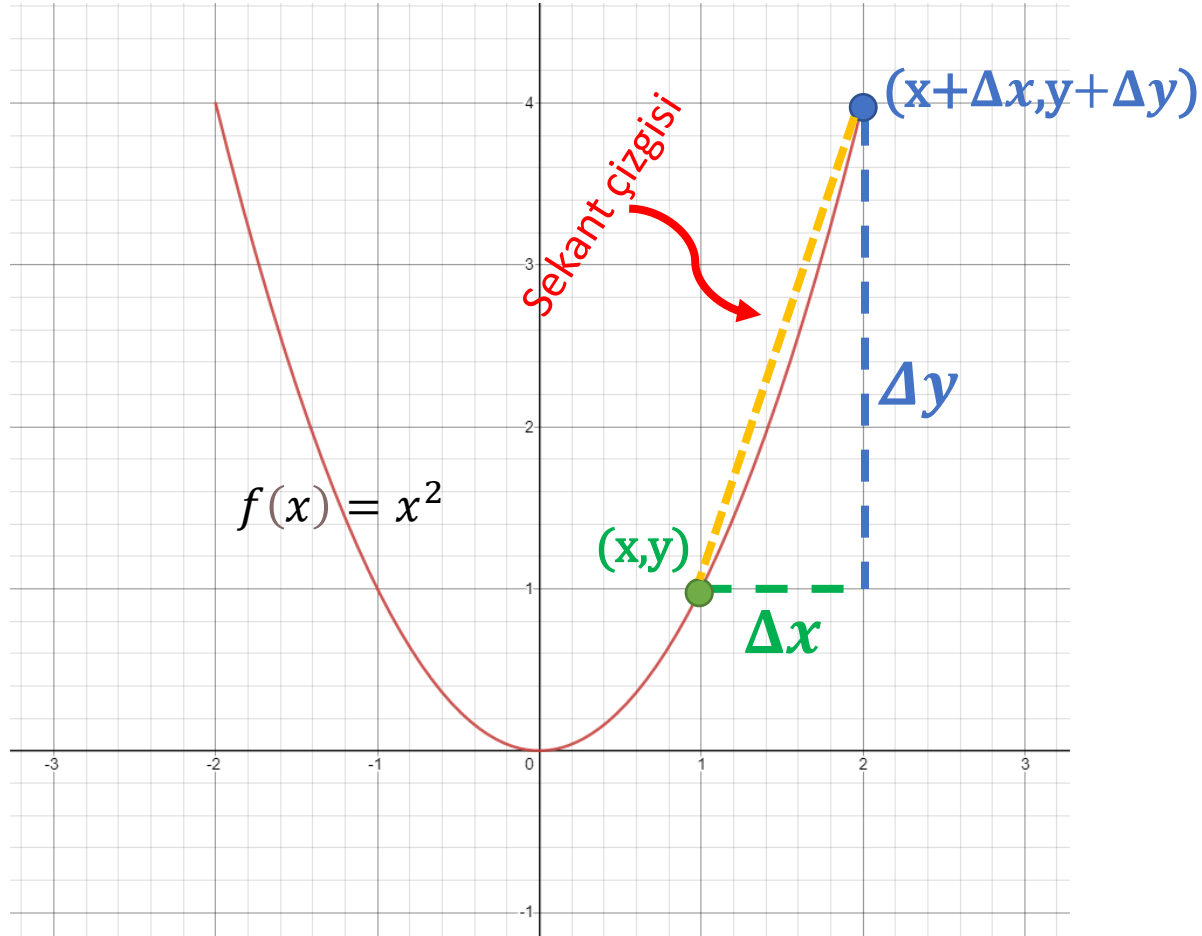
$$\text{eğim} = \frac{\text{y'deki değişim}}{\text{x'deki değişim}}$$

$$m = \frac{\Delta y}{\Delta x} = \frac{y_2 - y_1}{x_2 - x_1}$$

$$m = \frac{f(x + \Delta x) - f(x)}{(x + \Delta x) - x}$$

$$f'(x) = \lim_{h \rightarrow 0} \frac{f(x + h) - f(x)}{(x + h) - x}$$

$$f'(x) = \lim_{h \rightarrow 0} \frac{f(x + h) - f(x)}{h}$$



$$f'_{(x)} = \lim_{h \rightarrow 0} \frac{f(x + h) - f(x)}{h}$$

$$f'_{(x)} = \lim_{h \rightarrow 0} \frac{x^2 + 2xh + h^2 - x^2}{h}$$

$$f'_{(x)} = \lim_{h \rightarrow 0} \frac{2xh + h^2}{h}$$

$$f'_{(x)} = \lim_{h \rightarrow 0} 2x + h$$

$$f'_{(x)} = 2x$$

$$f'_{(x)} = nx^{n-1}$$

$\frac{\partial r}{\partial y}$ y'nin r deęiřimi

$$r = f(x, y) = x^2 + xy + y^2$$

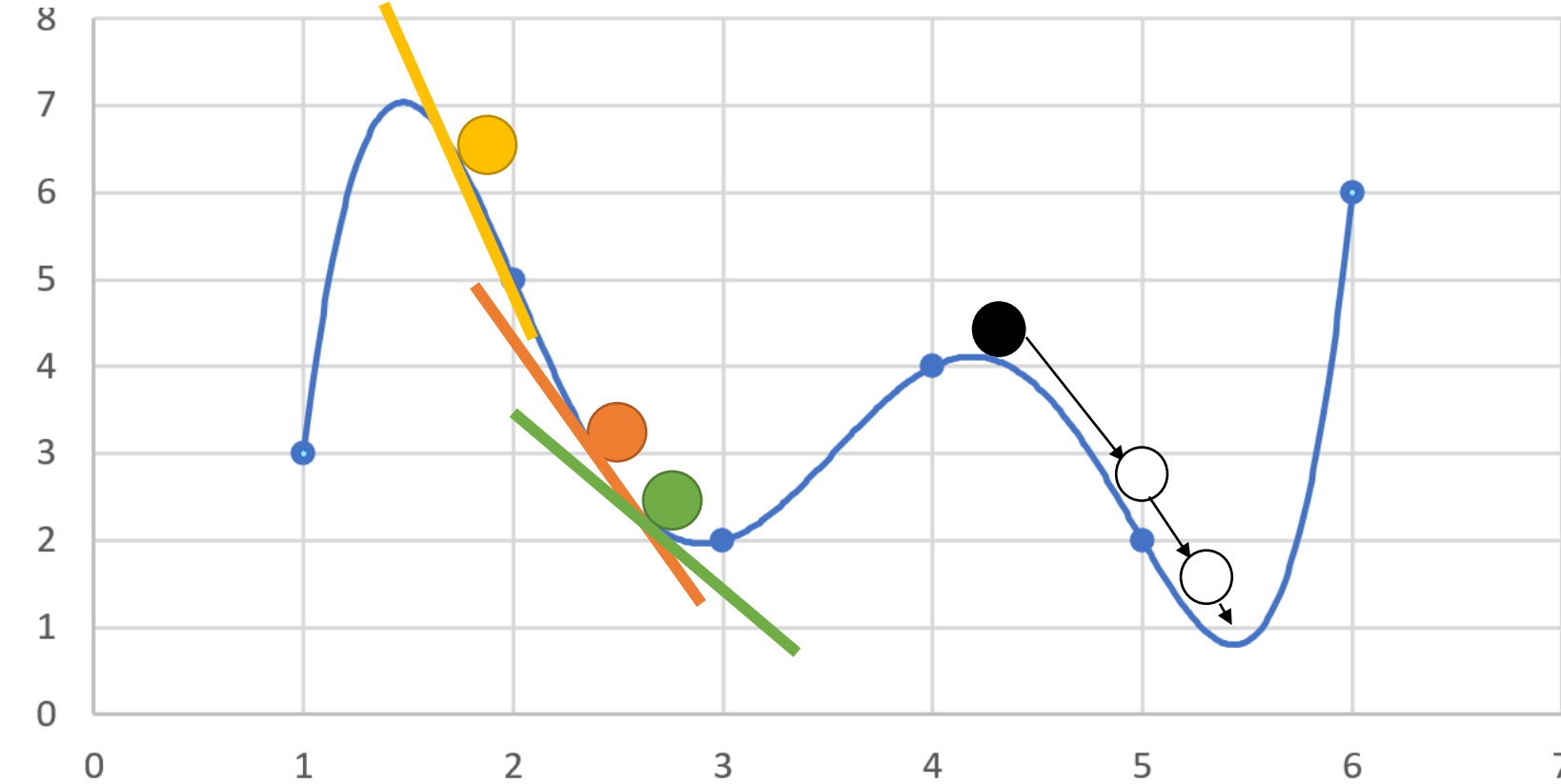
$$\frac{\partial r}{\partial x} = 2x + y$$

$$\frac{\partial r}{\partial y} = 2y + x$$

Sinir aęındaki hatanın aęırlıklara göre kısmi türevi alınır.

$$(f \circ g)'(t) = f'(g(t)) g'(t)$$

Tüm ağda türevlerin hesaplanmasında kısmi türevler önem arz etmektedir.



Dereceli alçalma

bir fonksiyonun yerel minimumunu bulmak için kullanılan yinelemeli bir optimizasyon algoritmasıdır.



1847 Baron Augustin-Louis Cauchy

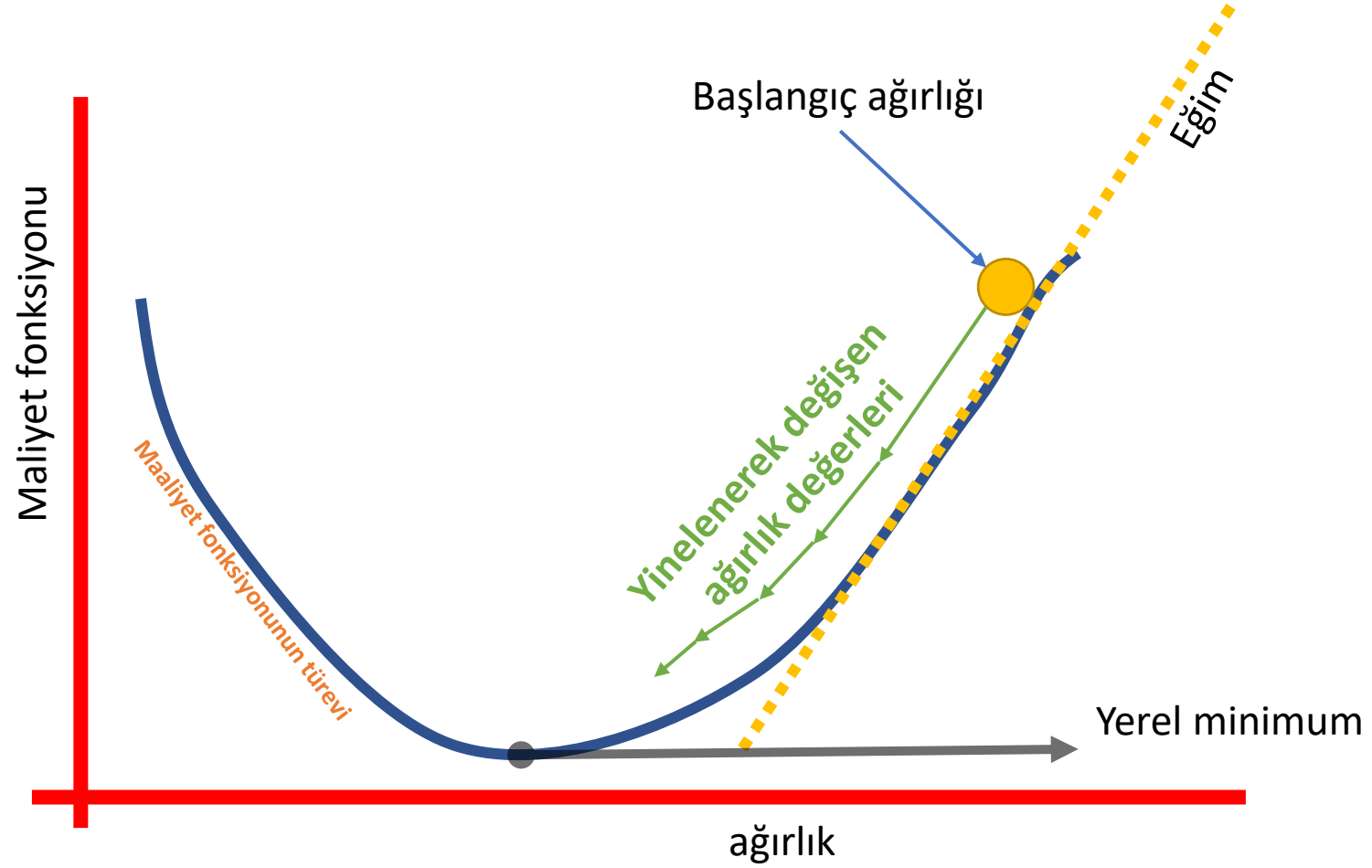
Dereceli alçalma genellikle Cauchy'ye atfedilir.
en dik iniş

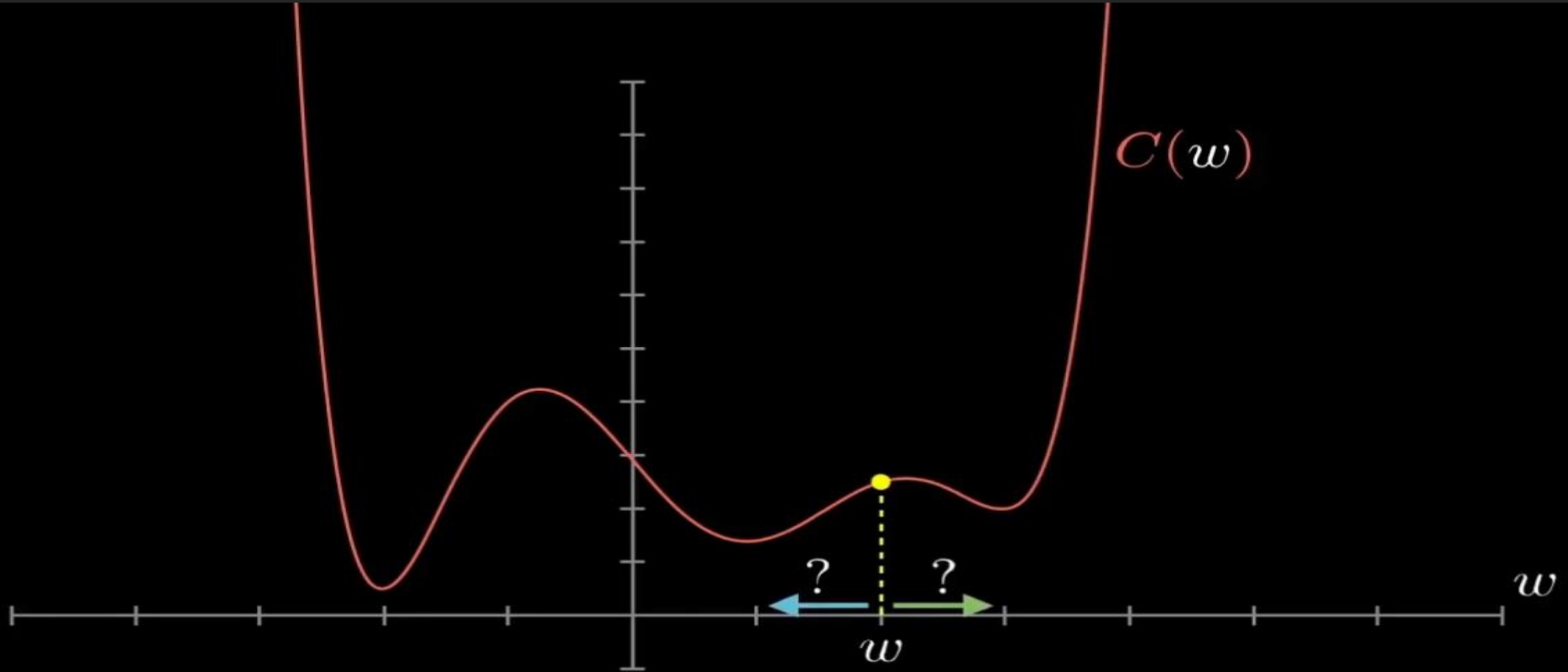


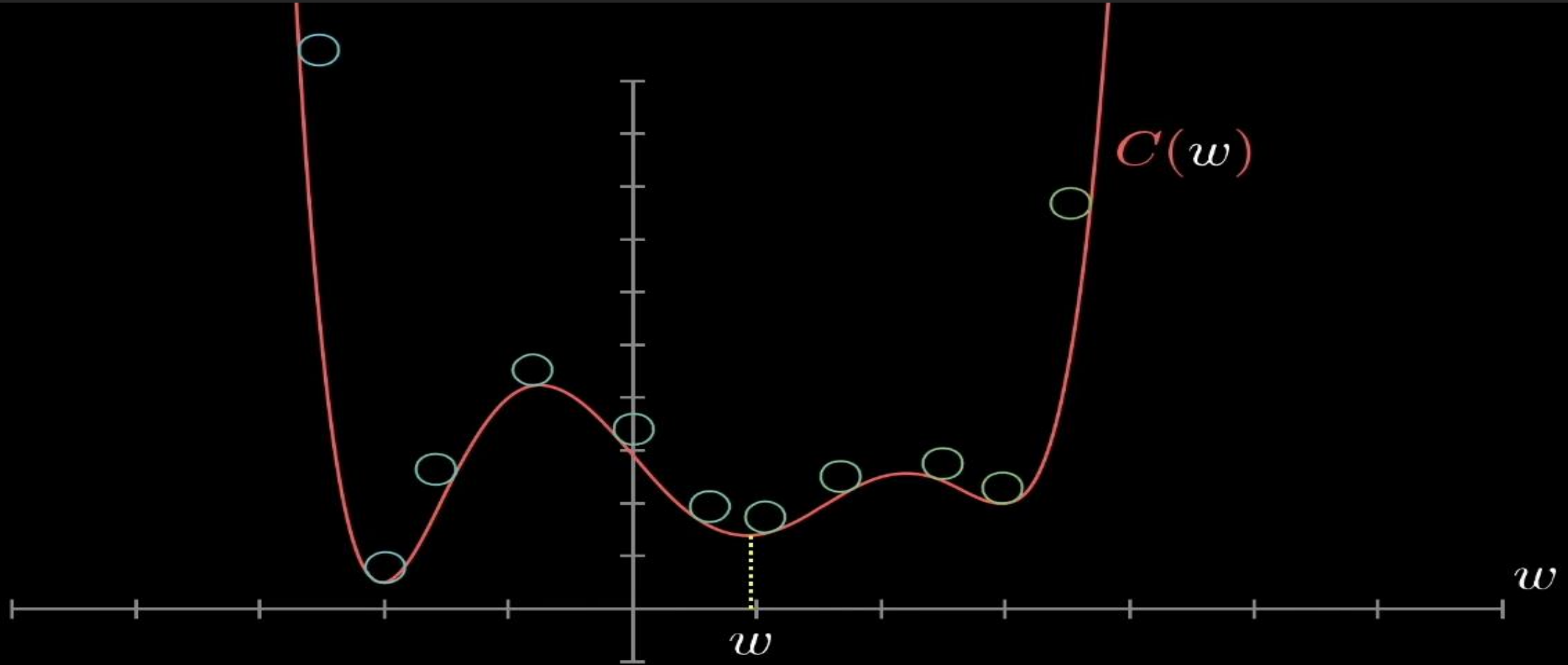
1944 Haskell Curry

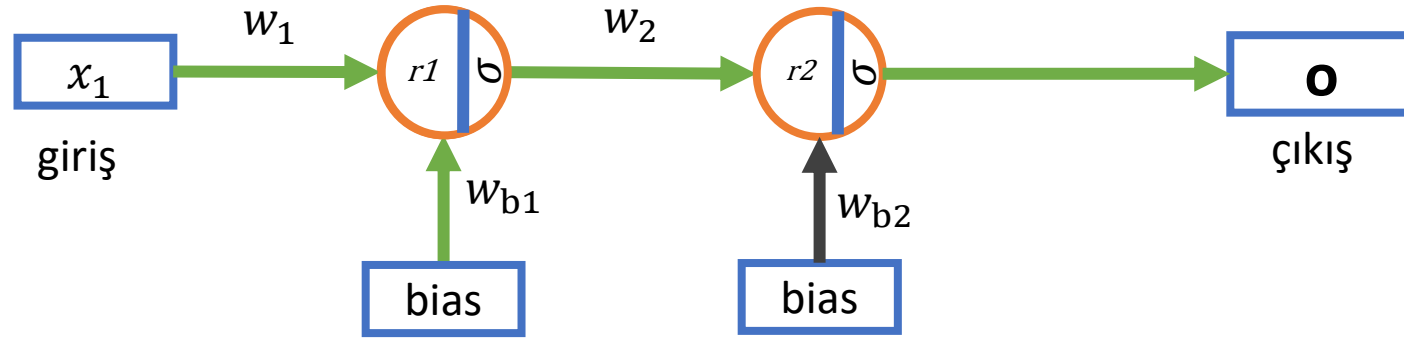
Doğrusal olmayan optimizasyon problemleri için yakınsama özelliklerini incelenmiştir.

Dereceli alçalma?
gradient decent?









Arzu edilen çıkış

d

$$Hata = o - d$$

$$C_0 = (o - d)^2$$

Cost
maliyet

iteration
adım

$$o = \sigma(\underbrace{n_1 \cdot \omega_2 + \omega_{b2}}_{r_2})$$

Aktivasyon fonksiyonu

$$o = \sigma(r_2)$$

$$\frac{\partial C_0}{\partial \omega_2} = \frac{\partial r_2}{\partial \omega_2} \frac{\partial \sigma}{\partial r_2} \frac{\partial C_0}{\partial \sigma}$$

Zincir kuralı

Dereceli alçalma?
gradient decent?

$$C_0 = (o - d)^2 \quad \Bigg| \quad o = \sigma(\underbrace{n_1 \cdot \omega_2 + \omega_{b2}}_{r_2}) \quad \Bigg| \quad o = \sigma(r_2) \quad \Bigg| \quad \frac{\partial C_0}{\partial \omega_2} = \frac{\partial r_2}{\partial \omega_2} \frac{\partial \sigma}{\partial r_2} \frac{\partial C_0}{\partial \sigma}$$

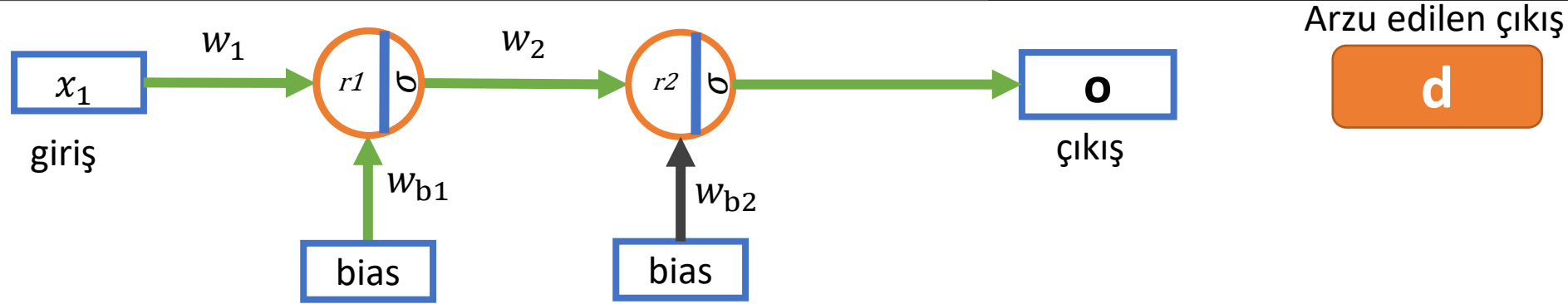
$$\frac{\partial C_0}{\partial \sigma} = 2 * (o - d)$$

$$\frac{\partial \sigma}{\partial r_2} = \sigma'(r_2)$$

$$\frac{\partial r_2}{\partial \omega_2} = n_1$$

$$\frac{\partial C_0}{\partial \omega_2} = (2 * (o - d)) * (\sigma'(r_2)) * (n_1)$$

$$\frac{1}{n} \sum_{i=1}^n \frac{\partial C_i}{\partial \omega_2}$$

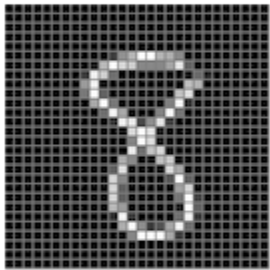


$$\nabla C^T = \left[\frac{\partial C}{\partial \omega_1}, \frac{\partial C}{\partial \omega_{b1}}, \frac{\partial C}{\partial w_2}, \frac{\partial C}{\partial \omega_{b2}} \right]$$

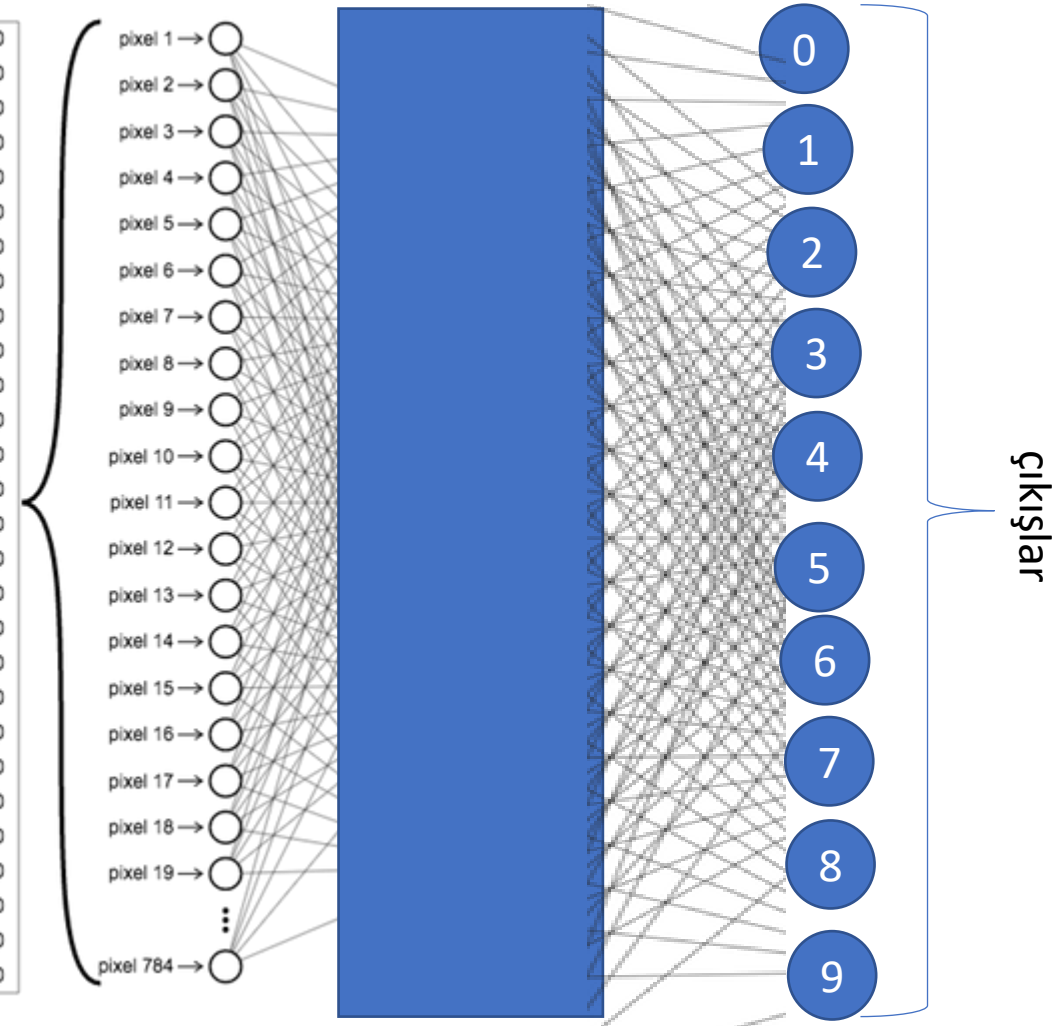
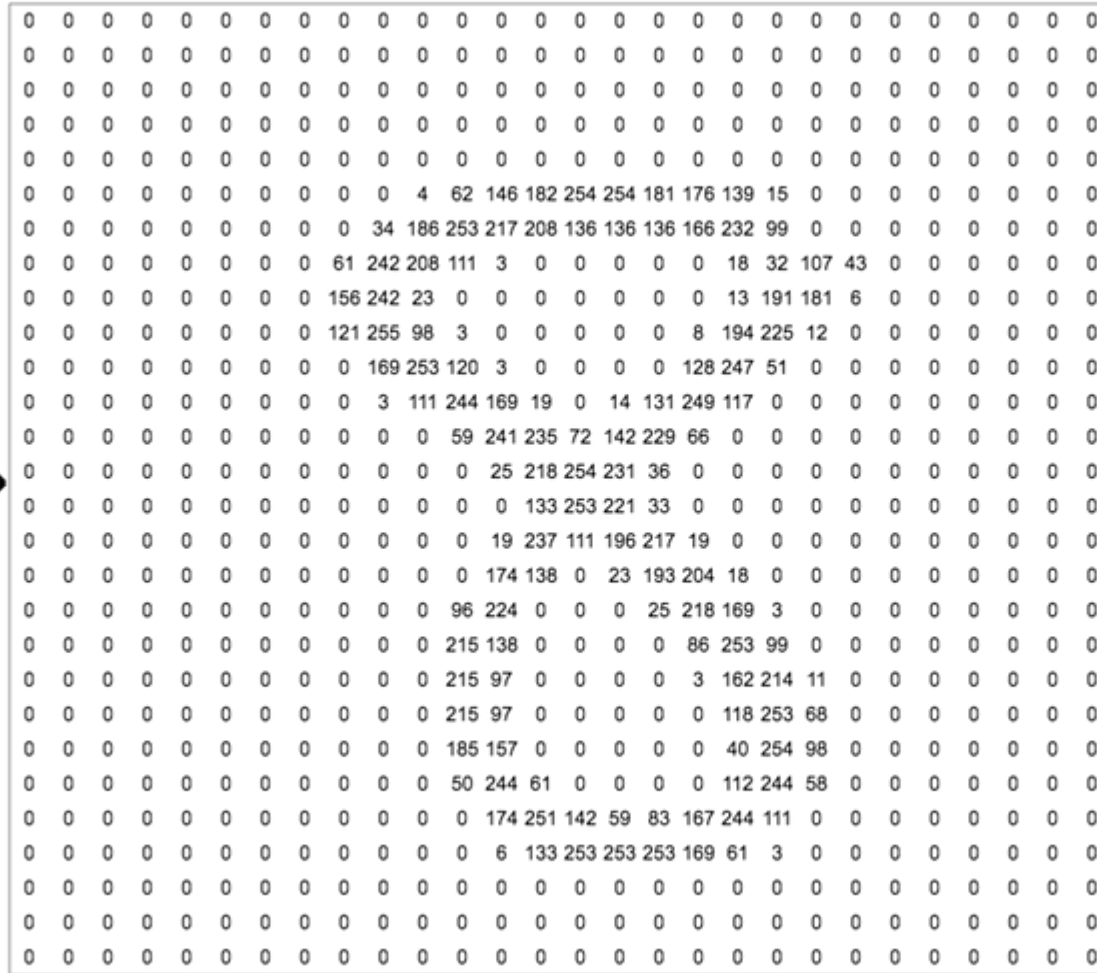


geri yayılımbackpropagation

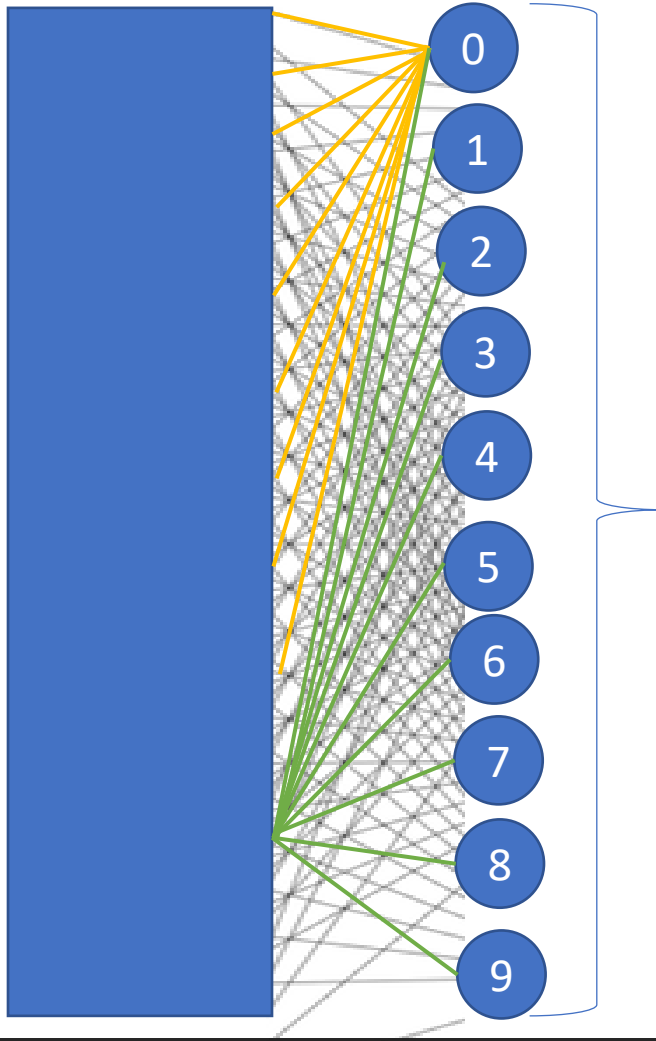
Mnist el yazısı rakamlar data seti



28 x 28
784 pixels



back propagation ?



	0	1	2	3	4	5	6	7	8	9	
w1	↑	↑	↓	↑	↓	↑	↓	↓	↓	↑	↓
w2											
w3	↓	↓	↑	↓	↑	↓	↑	↑	↑	↓	↑
.											
.											
.											
.											
wn											

Mini-batch Küçük grup
Küçük yığın

1. Grup seç
2. Grubu YSA da işle
3. Küçük grubun ortalama gradyanını bul
4. Ağırlıkları bulunan ortalama gradyanla güncelle
5. Gruplar bitene kadar işleme devam devam et

