Dersin içeriği

- Yapay Zeka
- Makine Öğrenmesi
- **Derin Öğrenme**
- Nesnelerin İnterneti
- Dijital Dönüşüm
- Artırılmış ve Sanal Gerçeklik
- Dijital vatandaşlık
- Robotik

Derin öğrenme

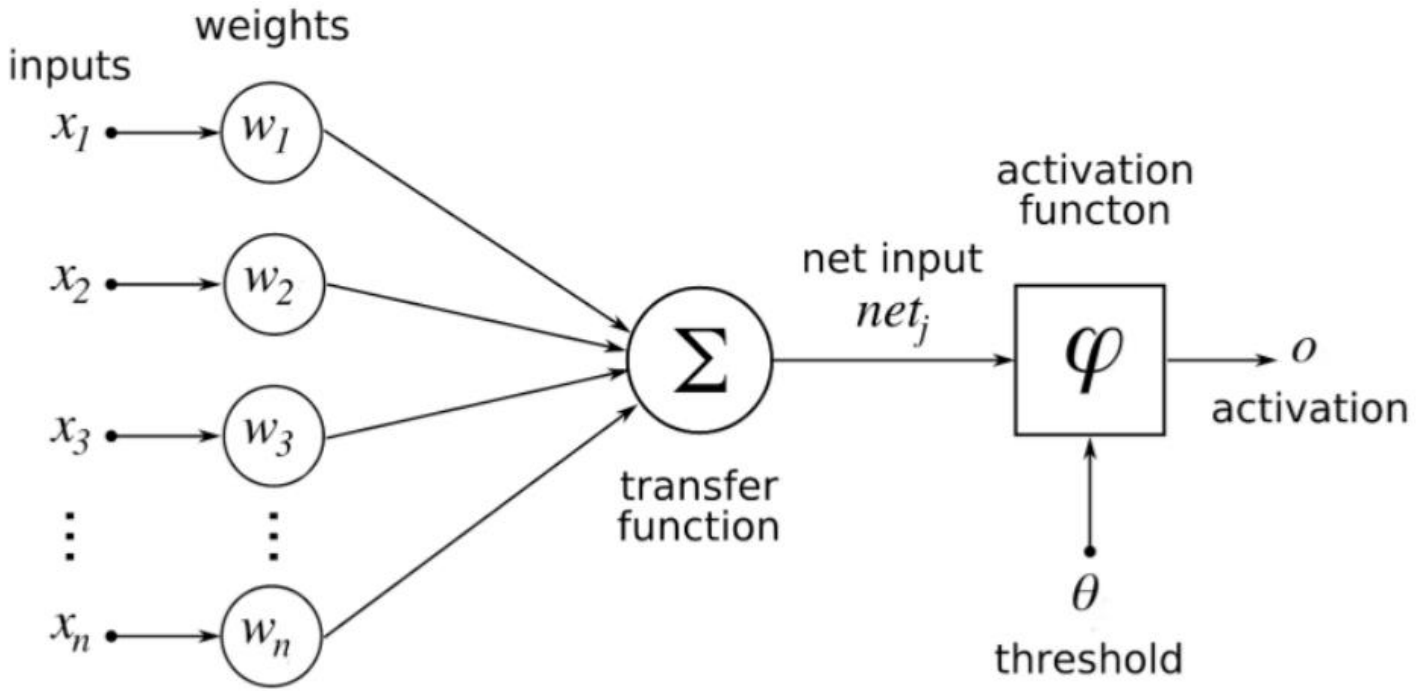
- Nedir bu “derin öğrenme”?
- Neden her yerde “derin öğrenme” görüyoruz?
- Derin öğrenme modellerinin klasik yapay sinir ağlarından farkı?
- Bu modellerin uygulamaları nelerdir?
- Derin öğrenme modelleri nasıl eğitilir?
- Nereden başlayabilirim?

Derin öğrenme

- **“Derin öğrenme” nedir?**
- Makine öğrenmesinin bir alt dalı.
- Beynin yapısal ve işlevsel özelliklerinden esinlenilerek tasarlanmış, çok katmanlı ağ yapıları olan “yapay sinir ağları” üzerinde çalışan algoritmalar ve modeller kümesi.

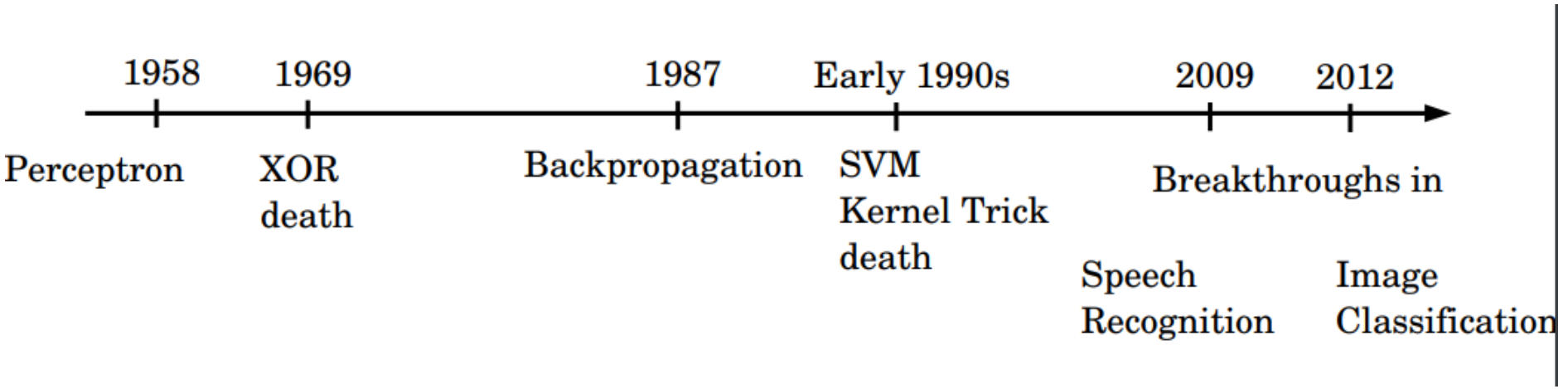
Derin öğrenme

- Perceptron: bir yapay nöron modeli



$$o=\varphi\left(\sum_{i=1}^{D} w_i x_i;\theta\right) \text{ where } x\in\mathbb{R}^D, o\in\mathbb{R}$$

Derin öğrenme



- 2009'da G. Hinton ve öğrencileri konuşma tanıma problemi için yeni bir eğitme yöntemi geliştirdi.
  Eğitmensiz (unsupervised) öğrenme ile ağı ilklediler.
  En sona "eğitmenli" katmanı ekleyip geriyayılım kullandılar.
  Bu yöntemle uzun süredir en iyi sonucu veren modeli geçtiler.
  Yöntemleri Android telefonlarda 2012'den itibaren kullanılmaya başlandı (DBN).

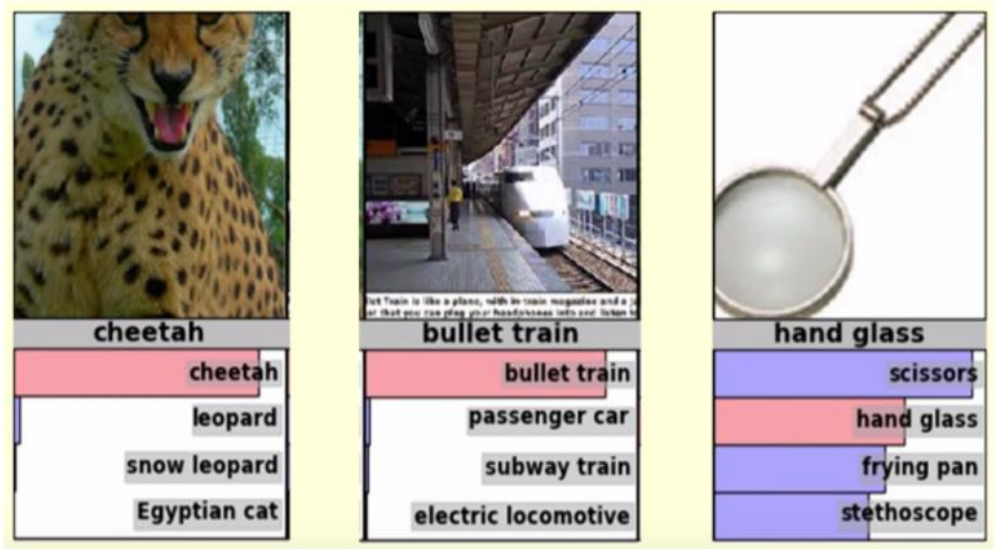
Derin öğrenme

- 2012 ILSVRC yarışması

1,2 milyon görüntü, 1000 sınıf, verilen görüntü için, o görüntüdeki baskın nesneyi tahmin etmeye çalışın.

5 tahmin üretin. Bu 5 tahminden biri doğruysa, başarılı sayılıyor.

G. Hinton ve öğrencisi Alex Krizhevsky, 2009'daki yöntemi kullanarak 7 katmanlı bir evrişimsel sinir ağı eğitti (convolutional neural network) Bu ağ, günümüzde "AlexNet" olarak biliniyor.



Derin öğrenme

- Derin öğrenme modellerinin klasik yapay sinir ağlarından ne farkı var?
- “Temelde hiçbir farkları yok” demek yanlış olmaz.
- Yeni olan şeyler:
  - Daha çok veri ve daha çok işlem gücü
  - Yeni nonlineer aktivasyon fonksiyonları
  - Yeni initialization yöntemleri
  - Yeni regularization yöntemleri

Derin öğrenme

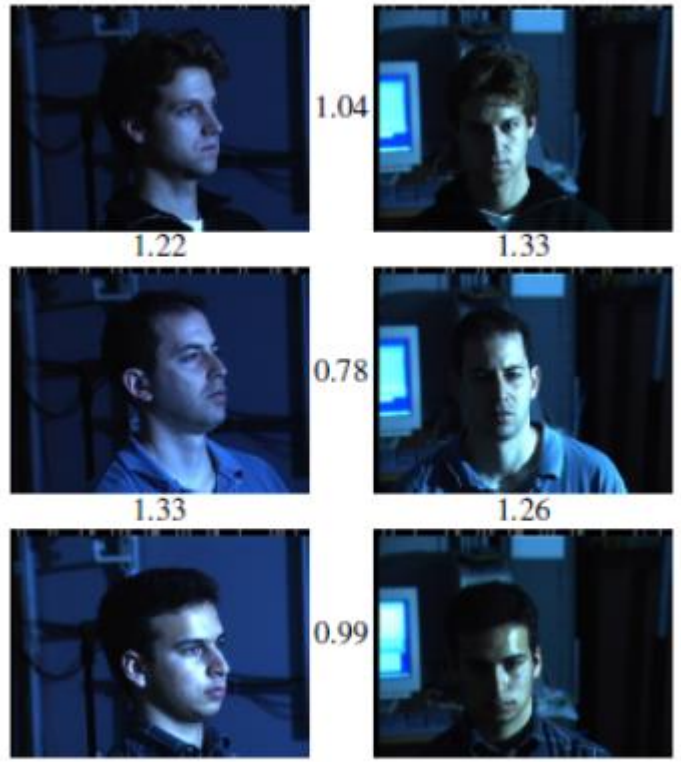
- Yüz tanıma:
- Yüz tanıma iki şekilde yapılabilir.
- Bunu yapmanın ilk yolu, yüzleri doğru bir şekilde sınıflandırabilen bir nöral ağ modeli (tercihen bir ConvNet modeli) eğitmektir.
- Bir sınıflandırıcının iyi eğitilmesi için milyonlarca giriş verisine ihtiyaç vardır. Bir işletme için çalışanların bu kadar çok fotoğrafını toplamak çoğu zaman mümkün değildir. Yani bu yöntem çok kullanışlı değildir.
- Bunu yapmanın ikinci yolu ise tek adımda öğrenme tekniğini (one-shot learning) seçmektir.
- Tek adımda öğrenme, nesne kategorizasyon problemleri için kullanılan bir bilgisayarlı görü yöntemidir. Bir veya yalnızca birkaç eğitim görüntüsünden bilgi öğrenmeyi amaçlamaktadır. Modelin hala milyonlarca veri üzerinde eğitilmesi gerekiyor, ancak veri kümesi aynı alandan herhangi biri olabilir.
- Tek adımda öğrenme yolunda, herhangi bir yüz veri kümesiyle bir modeli eğitebilir ve daha az sayıda olan kendi verileriniz için kullanabilirsiniz.

Derin öğrenme

- Tek adımda öğrenme bir Siamese ağı kullanılarak uygulanabilir. Siamese ağı aynı ağırlıklara sahip, ancak iki farklı girdi alan iki özdeş sinir ağı durumudur. Bu ağlar, çıkışları arasındaki zıt kayıplara göre optimize edilir. Ağlara girişler benzer olduğunda, bu kayıp küçük olur ve girişler birbirinden farklı olduğunda ise büyük olur. Bu şekilde, optimize edilmiş Siamese ağları girişleri arasında ayrım yapabilir.
- Günümüzde yüz tanıma sistemleri tek adımda öğrenme tekniğini tercih etmektedir.

Derin öğrenme

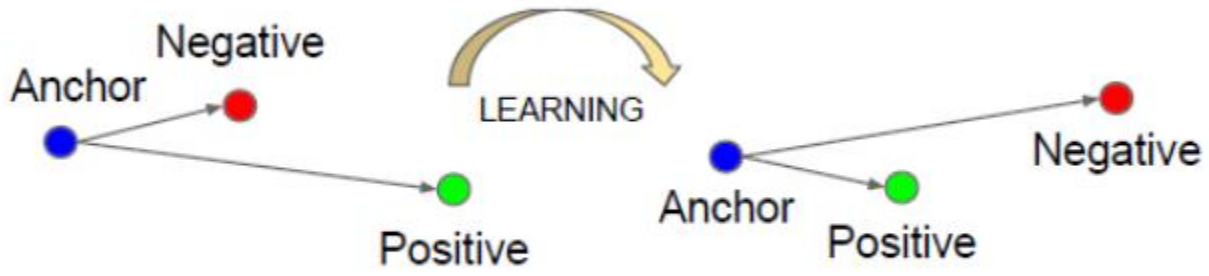
- Yüz tanıma: FaceNet
- *FaceNet*, yüz tanıma, doğrulama ve kümeleme gibi görevleri gerçekleştirmek için başarılı bir mimari sağlar.
- 140 milyon parametreden oluşan *FaceNet*, girdi olarak aldığı yüz resmini en iyi şekilde ifade etmeye çalışan 128 boyutlu bir özellik vektörü üretir.
- Şekil 1'de resimlerden elde edilen vektörlerin benzerliği gösterilmektedir. Görüldüğü üzere benzer resimlerin özellik vektörlerinin benzerliği yüksek iken farklı resimlerin özellik vektörlerinin benzerliği düşüktür.
- FaceNet, en iyi özellik vektörünü üretebilmek için üçlü kayıp (triplet loss) ile birlikte evrişimli sinir ağlarını (CNN) kullanır.



Şekil 1 — Yüz vektörlerinin öklid benzerliği [4]

Derin öğrenme

- Bir derin öğrenme modeli eğitilirken öğrenmenin sağlanabilmesi için ileri yayılım sonucunda bir yitim (loss) değeri hesaplanır. Bu yitim değeri kullanılarak geri yayılım algoritması ile ağırlıklar güncellenir. Dolayısıyla Ağ'ın başarılı bir şekilde eğitilebilmesi için yitim fonksiyonu kritik bir öneme sahiptir. *FaceNet* mimarisinde kullanılan üçlü kayıp yitim fonksiyonu hesaplanırken veri kümesinden faklı üçlü gruplar seçilir ve üçlü kayıp değeri bu 3 resim üzerinden hesaplanır. Belirlenen bu üç resmin biri referans resim (anchor) iken, diğer ikisi referans resme benzeyen (pozitif) ve benzemeyen (negatif) resimlerdir.



Şekil 3 — Üçlü Kayıp (Triplet Loss) [4]

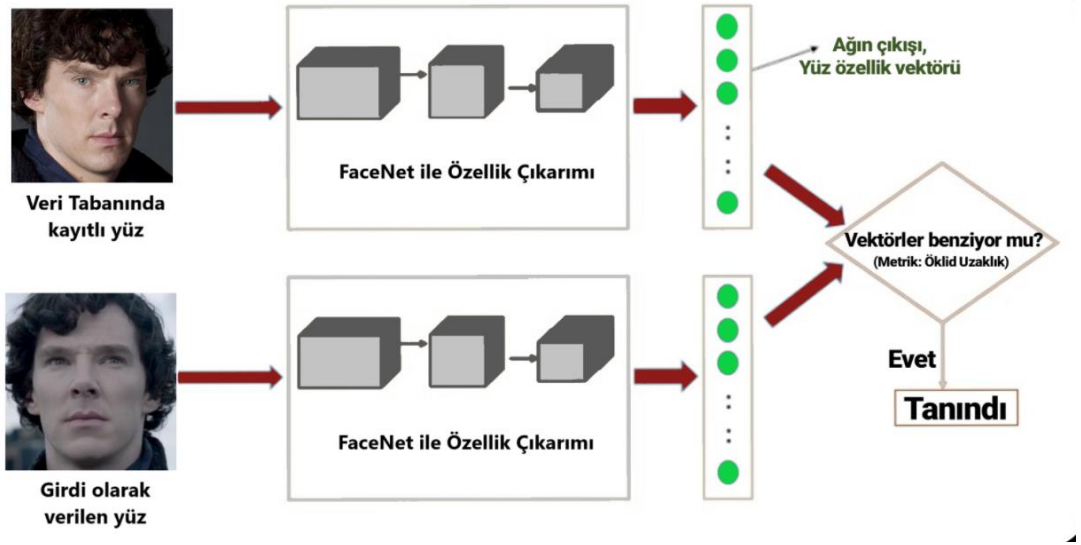
Derin öğrenme

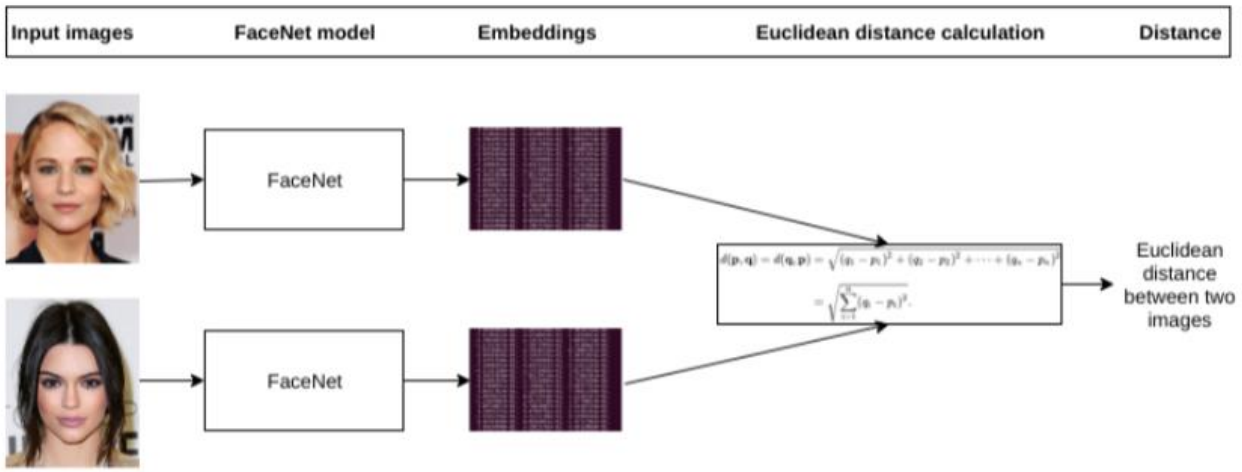
- Yitim değeri referans resim ile pozitif olarak belirlenmiş resim arasındaki benzerlik arttıkça küçülürken, negatif olarak belirlenmiş resim ile arasındaki benzerlik arttıkça büyür. Bu sayede ***öğrenme sonucunda referans resim vektörü pozitif resim vektörü ile birbirlerine benzerken negatif resim vektöründen farklılaşmaktadır***. Böylece eğitimi tamamlandıktan sonra *FaceNet*, girdi olarak aldığı resmi en iyi temsil eden özellik vektörünü çıkarmayı öğrenmiş olur.

layer	size-in	size-out	kernel	param	FLPS
conv1	220×220×3	110×110×64	7×7×3, 2	9K	115M
pool1	110×110×64	55×55×64	3×3×64, 2	0	
rnorm1	55×55×64	55×55×64		0	
conv2a	55×55×64	55×55×64	1×1×64, 1	4K	13M
conv2	55×55×64	55×55×192	3×3×64, 1	111K	335M
rnorm2	55×55×192	55×55×192		0	
pool2	55×55×192	28×28×192	3×3×192, 2	0	
conv3a	28×28×192	28×28×192	1×1×192, 1	37K	29M
conv3	28×28×192	28×28×384	3×3×192, 1	664K	521M
pool3	28×28×384	14×14×384	3×3×384, 2	0	
conv4a	14×14×384	14×14×384	1×1×384, 1	148K	29M
conv4	14×14×384	14×14×256	3×3×384, 1	885K	173M
conv5a	14×14×256	14×14×256	1×1×256, 1	66K	13M
conv5	14×14×256	14×14×256	3×3×256, 1	590K	116M
conv6a	14×14×256	14×14×256	1×1×256, 1	66K	13M
conv6	14×14×256	14×14×256	3×3×256, 1	590K	116M
pool4	14×14×256	7×7×256	3×3×256, 2	0	
concat	7×7×256	7×7×256		0	
fc1	7×7×256	1×32×128	maxout p=2	103M	103M
fc2	1×32×128	1×32×128	maxout p=2	34M	34M
fc7128	1×32×128	1×1×128		524K	0.5M
L2	1×1×128	1×1×128		0	
total				140M	1.6B

Şekil 2 — FaceNet Mimarisi

Derin öğrenme

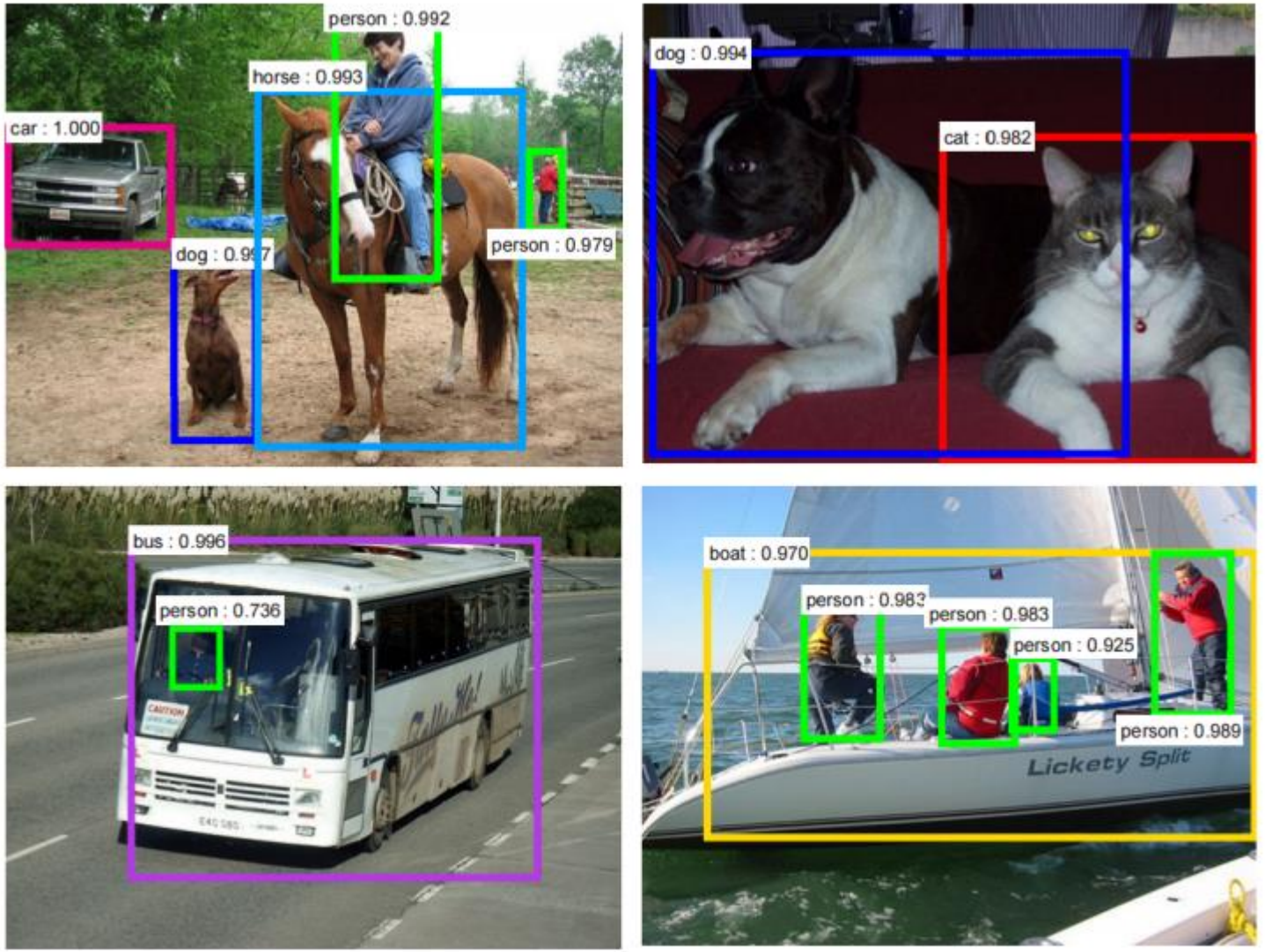




Bu bilgiler ışığında, iki yüzün aynı kişiye ait olup olmadığını tespit edeceğimizi varsayalım. ***Bu durumda Şekilde verildiği gibi girdi olarak aldığınız yüz ile veri tabanınızdaki yüz resminin FaceNet ile özellik vektörlerini elde etmemiz gerekir. Daha sonra bu iki vektörün benzerliğini bir metrik kullanarak ölçeriz. Benzerlik değeri daha önceden belirlediğimiz eşik değerinin altında ise bu durum kişilerin aynı kişi olduğu, üzerinde ise farklı kişi olduğu anlamına gelir.***

Derin öğrenme

- Nesne Algılama



Derin öğrenme

- Görüntü altyazılama

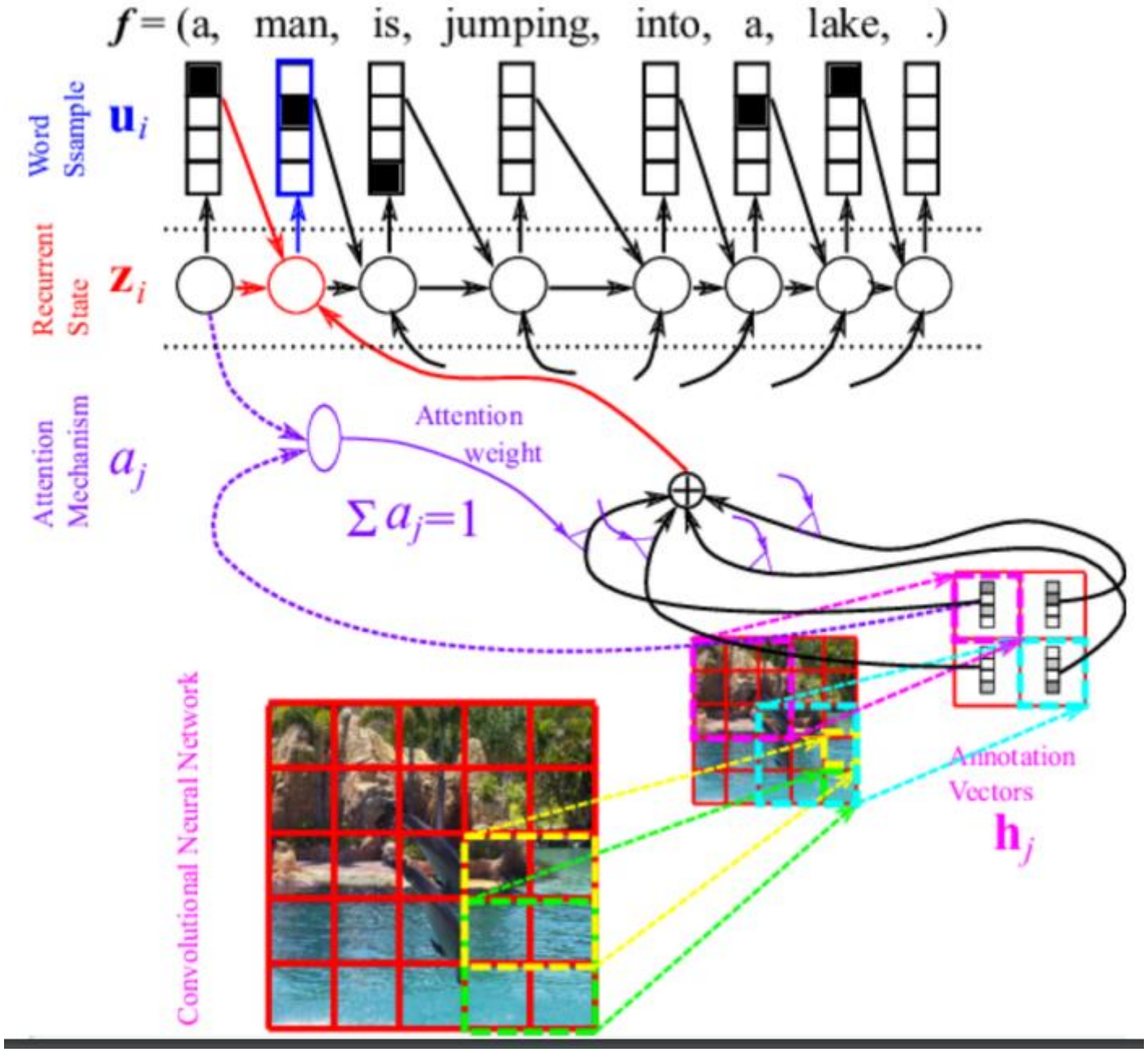
a street sign on a pole in front of a building

a plate with a sandwich and a salad

an elephant standing in a grassy field with trees in the background

Derin öğrenme

- Görüntü altyazılama

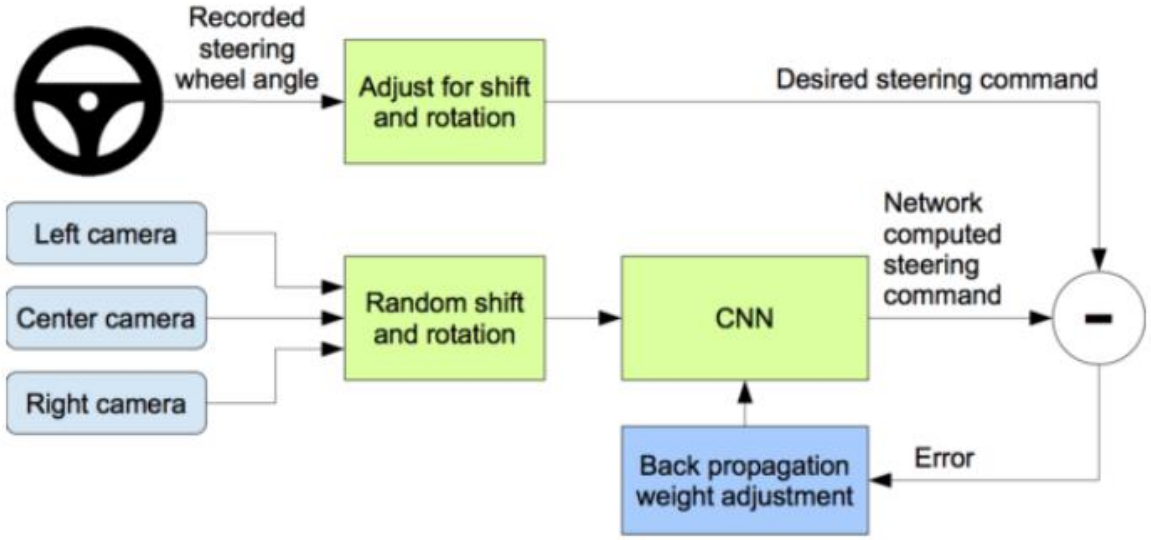


Derin öğrenme

- Derin üretici (generative) modeller

Derin öğrenme

- Sürücüsüz araçlar



Derin öğrenme

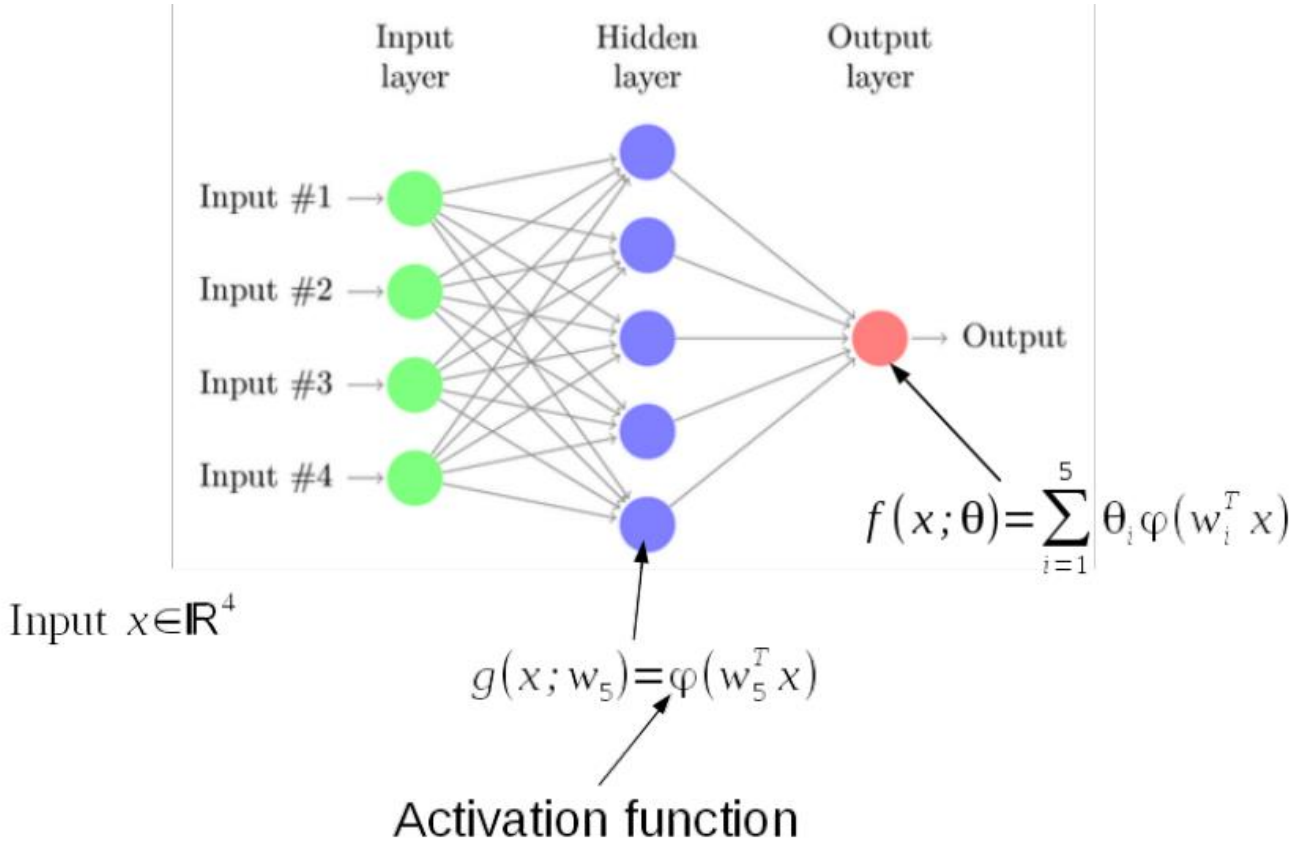
- Neden her yerde “derin öğrenme” görüyoruz?
- Birçok farklı problemde konvansiyonel makine öğrenmesi yöntemlerinden (çok) daha yüksek doğruluk verdiği için,
- Bu doğruluk seviyesinin ticari uygulamaları olanaklı kılmasından dolayı
- Yeni uygulamalara olanak sağladığı için.

Derin öğrenme

- Derin öğrenme modelleri nasıl eğitilir?
- İki ana derin model çeşidi:
- Çok katmanlı Perceptron (Multilayer Perceptrons)
- Evrişimsel Sinir Ağı (Convolutional Neural Networks)

Derin öğrenme

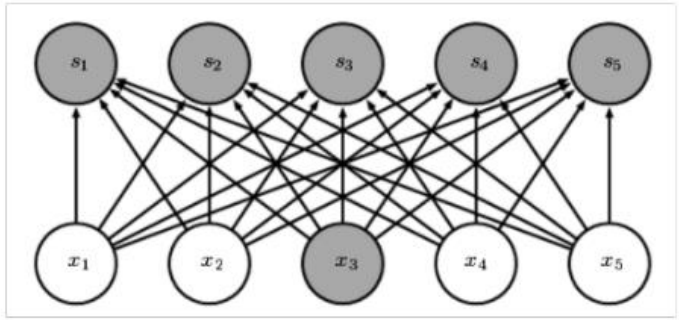
- Çok katmanlı perceptron



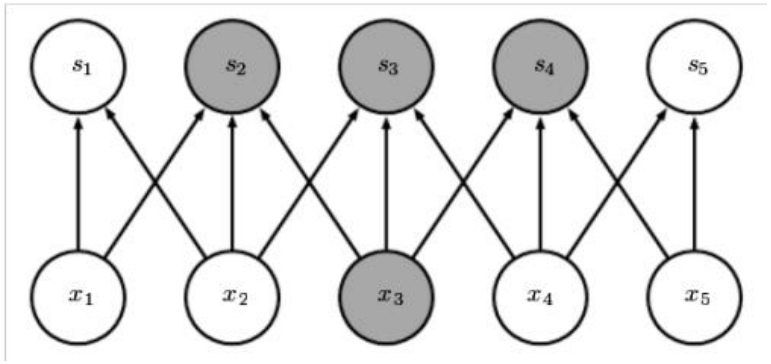
Derin öğrenme

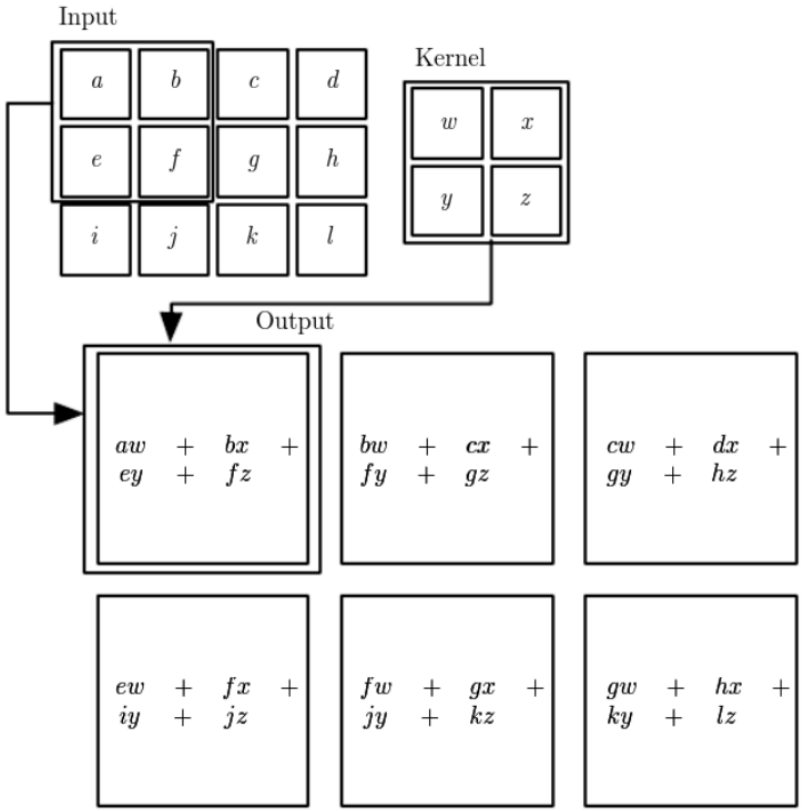
- Evrişimsel Sinir Ağları (Convolutional Neural Networks (CNNs))

In a usual ANN, nodes are fully-connected



In CNN, sparse connections:





Derin öğrenme

1- Verileri yükleme ve görselleştirme

2- Verilerin düzenlenmesi (reshaped, normalized, . Vb.)

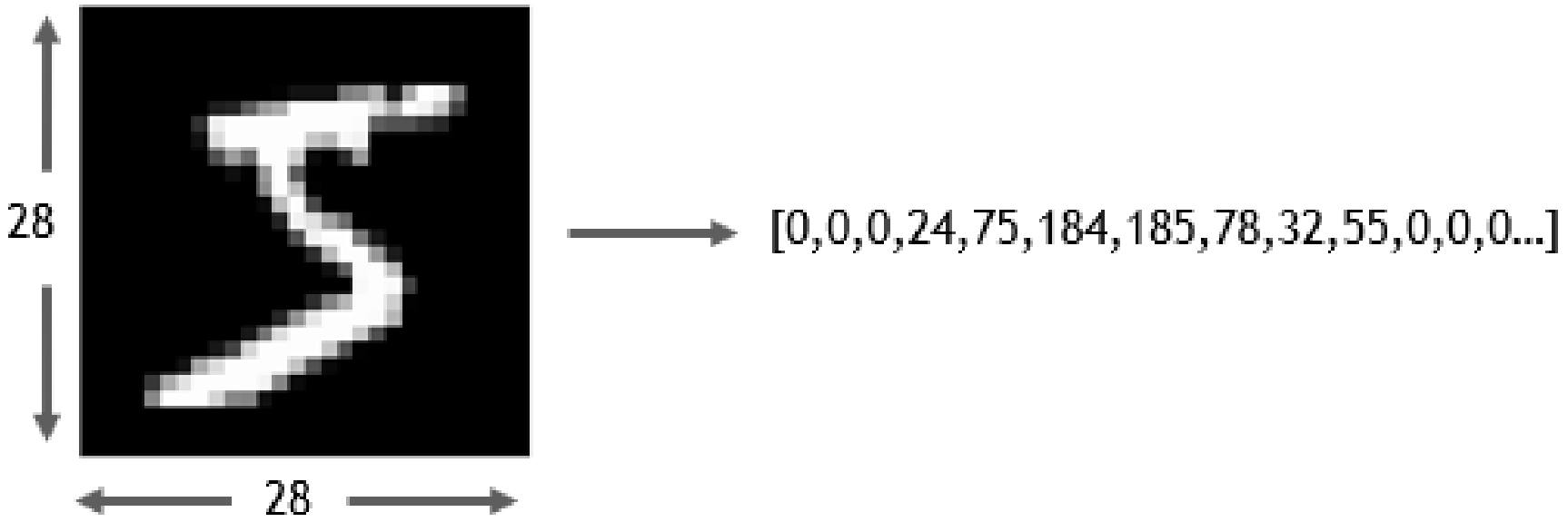
3-Modelin oluşturulması

4-Modelin çalıştırılması

5-Veriler üzerinde modelin eğitilmesi

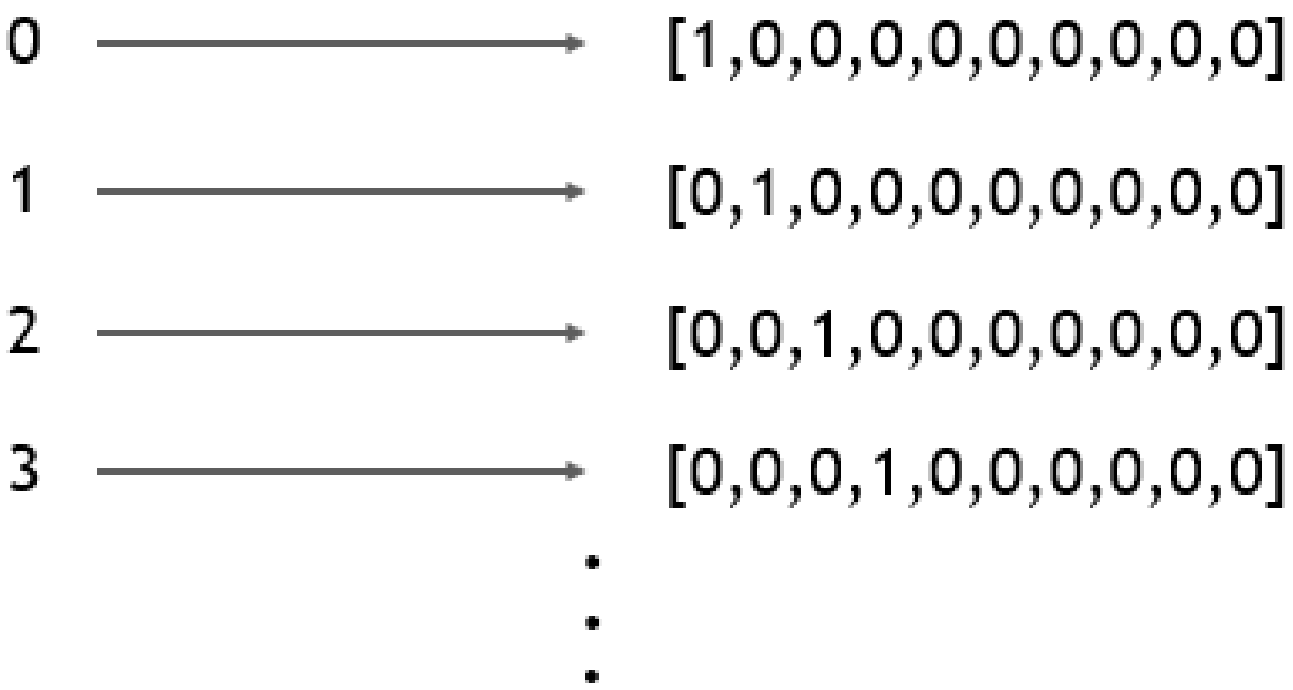
Derin öğrenme

1- Verileri yükleme ve görselleştirme

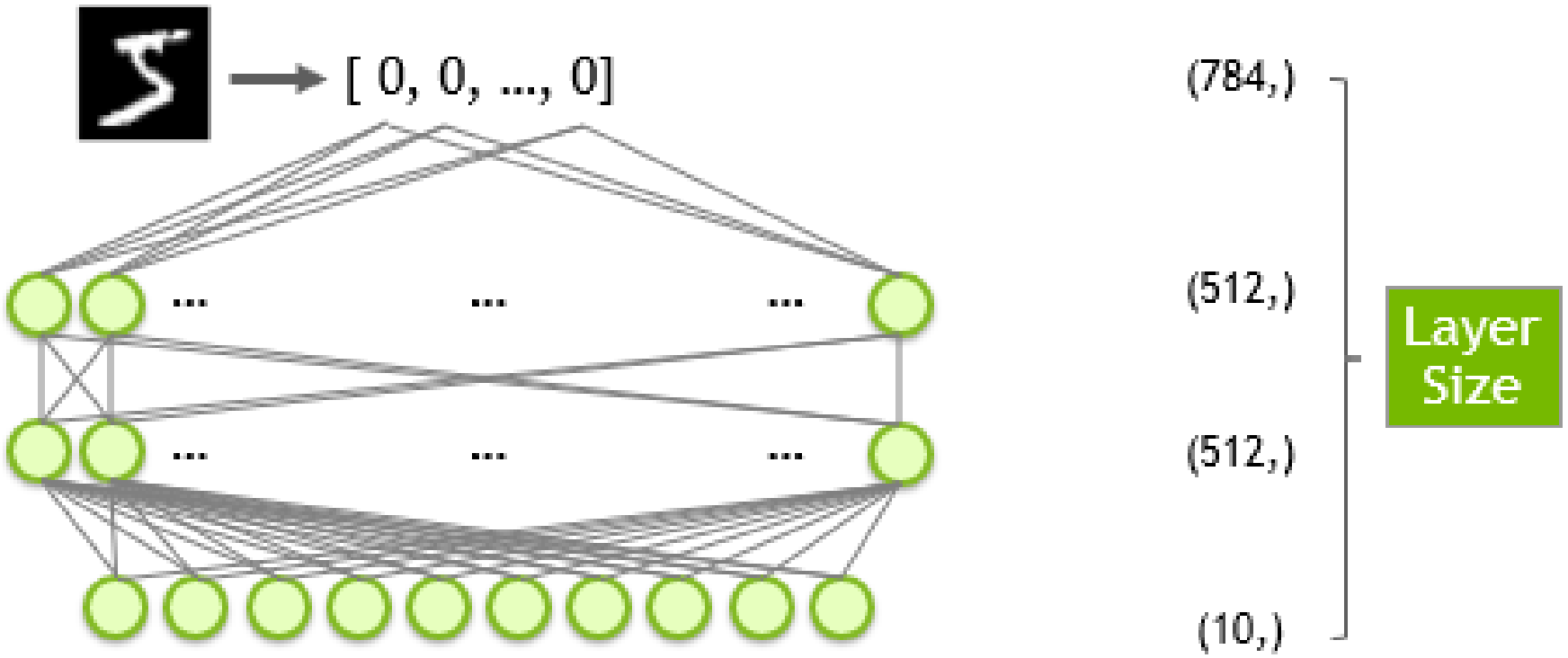


Derin öğrenme

2- Verilerin düzenlenmesi



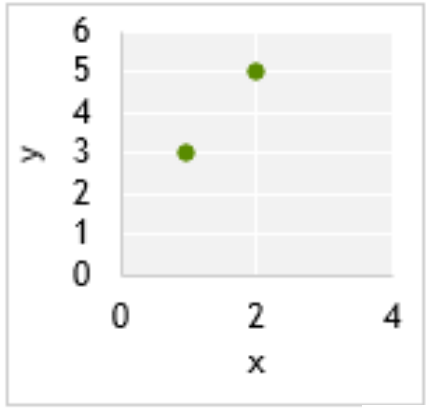
Derin öğrenme

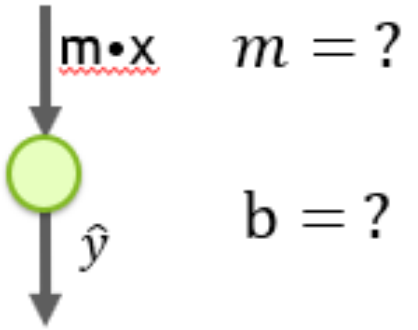


Derin öğrenme

- y=mx+b

x	y
1	3
2	5

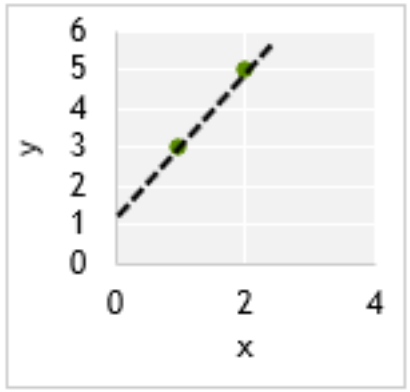


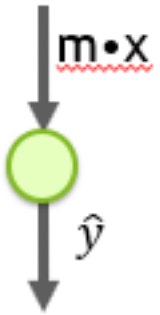


$m = ?$

$b = ?$

x	y
1	3
2	5





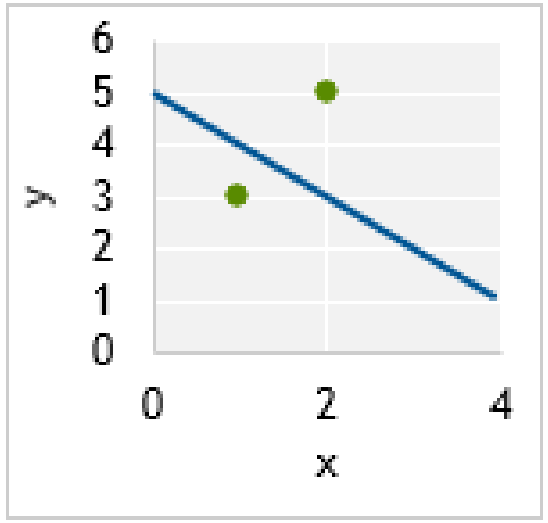
$m = ?$

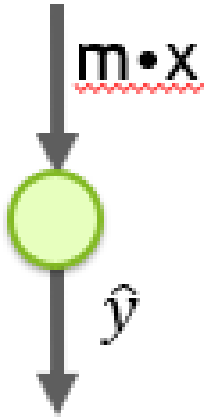
$b = ?$

Derin öğrenme

Basitçe, ortalama kare hata bir regresyon eğrisinin bir dizi noktaya ne kadar yakın olduğunu söyler.

x	y	$\hat{y}$
1	3	4
2	5	3





Start Random

$m = -1$

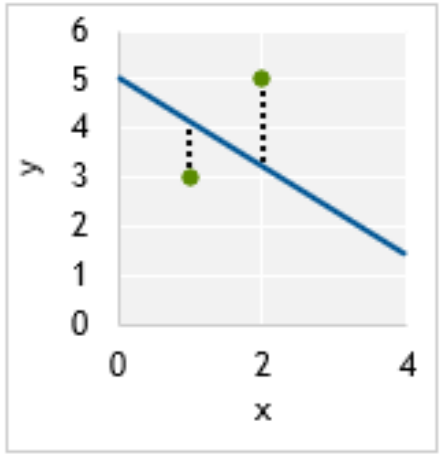
$b = 5$

Derin öğrenme

Mean Square Error (MSE): Bir makine öğrenmesi modelinin yani tahminleyicinin performansını ölçer, her zaman pozitif değerlidir ve MSE değeri sıfıra yakın olan tahminleyicilerin daha iyi bir performans gösterdiği söylenebilir.

Root Mean Square Error (RMSE): Tahminleyicinin tahmin ettiği değerler ile gerçek değerleri arasındaki uzaklığın bulunmasında sıklıkla kullanılan bir metriktir. Yani, regresyon hattının veri noktalarından ne kadar uzakta olduğunun bir ölçüsüdür. RMSE değeri 0'dan ∞'a kadar değişebilir. Negatif yönelimli puanlar yani daha düşük değerlere sahip tahminleyiciler daha iyi performans gösterir. RMSE değerinin sıfır olması modelin hiç hata yapmadığı anlamına gelir. RMSE, büyük hataları daha fazla cezalandırmanın avantajına sahiptir, bu yüzden bazı durumlara daha uygun olabilir.

x	y	$\hat{y}$	err^2
1	3	4	1
2	5	3	4
		MSE =	2.5
		RMSE =	1.6

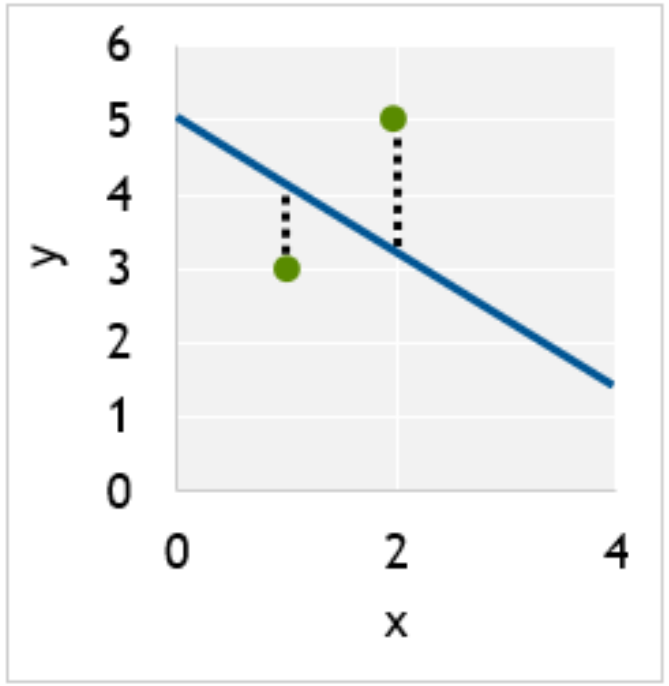


$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

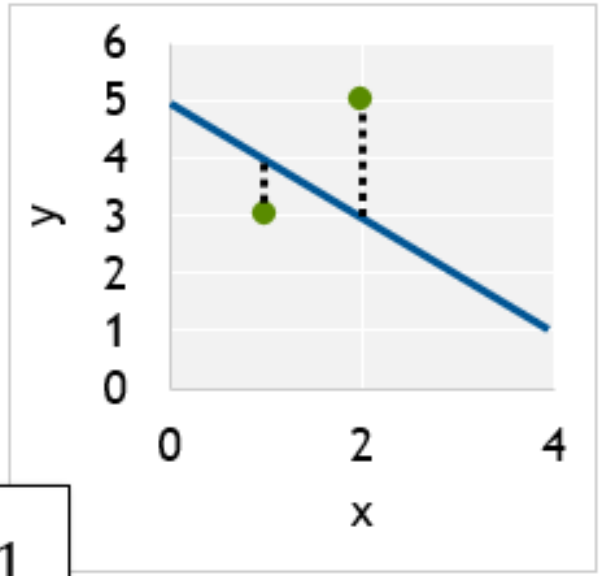
$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

Derin öğrenme

x	y	$\hat{y}$	err^2
1	3	4	1
2	5	3	4
		MSE =	2.5
		RMSE =	1.6

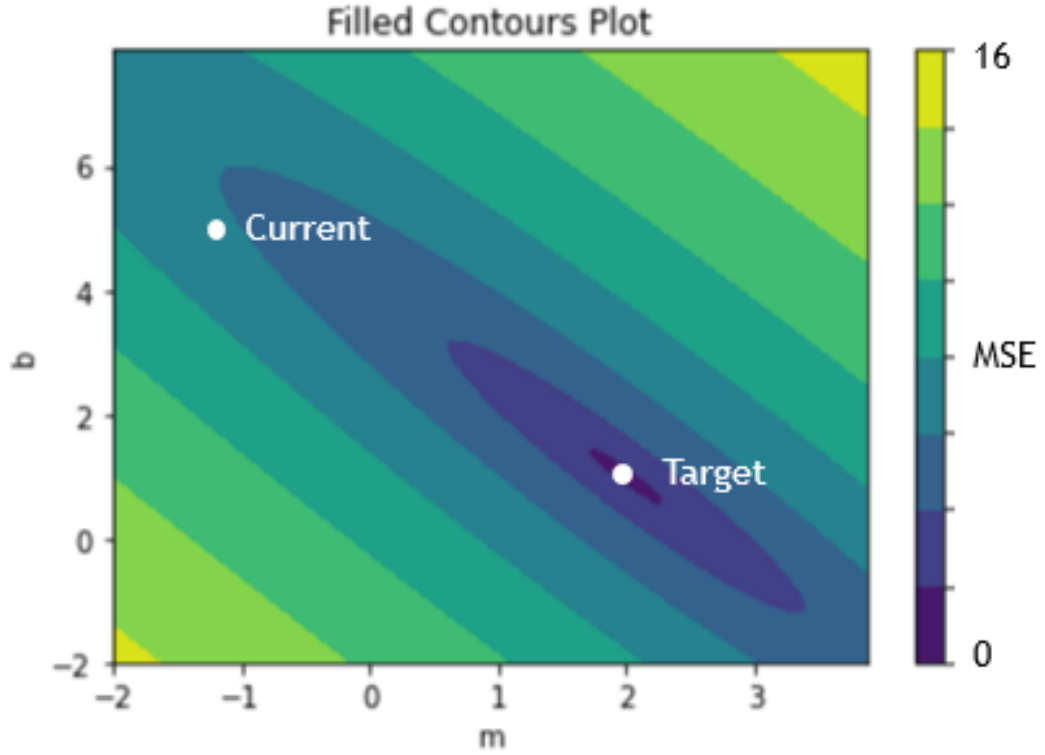


Derin öğrenme

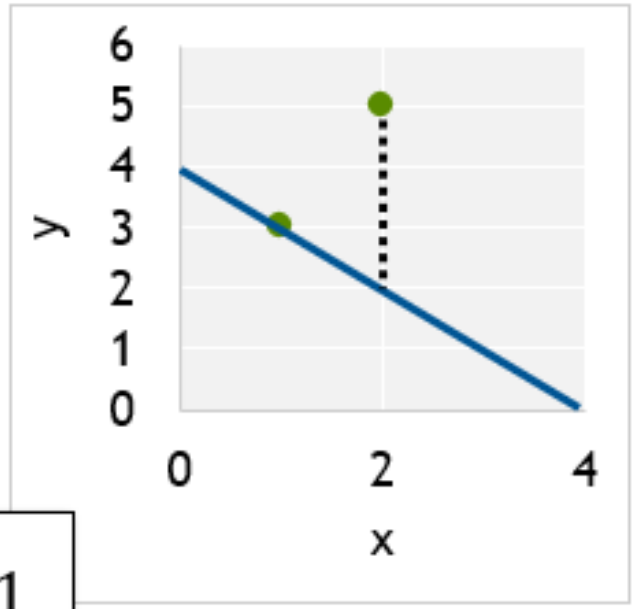


$m = -1$

$\mathrm{b} = 5$

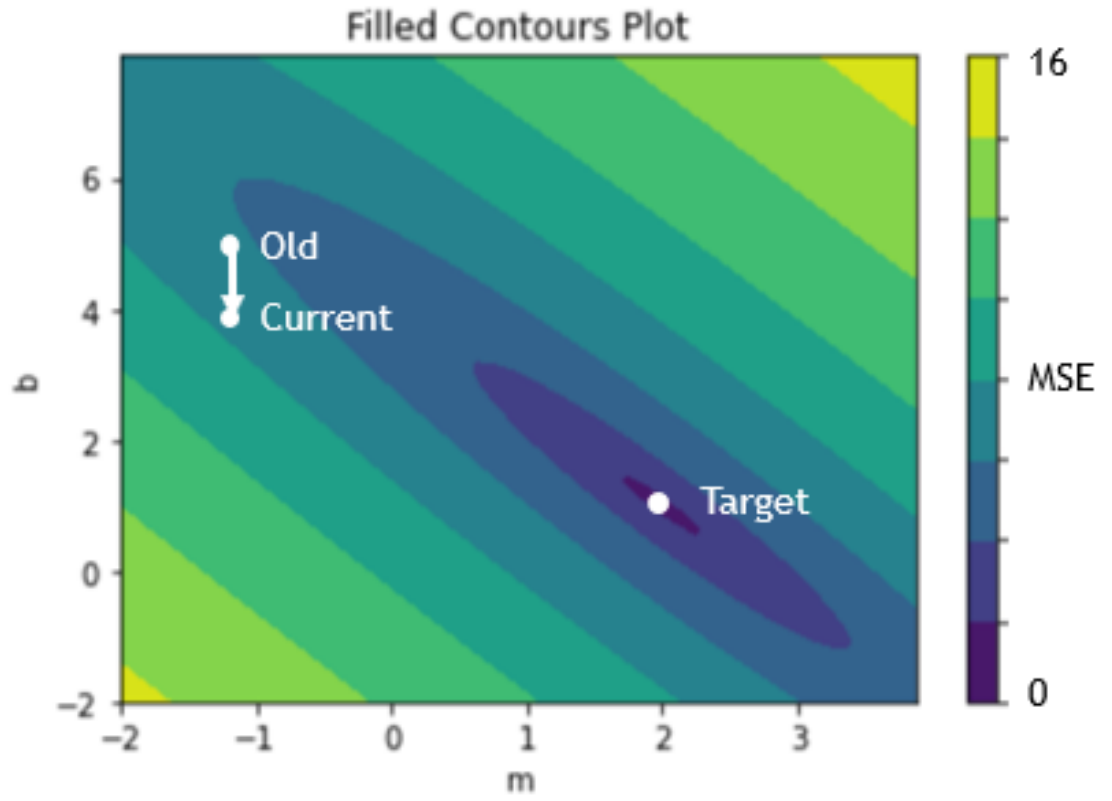


Derin öğrenme

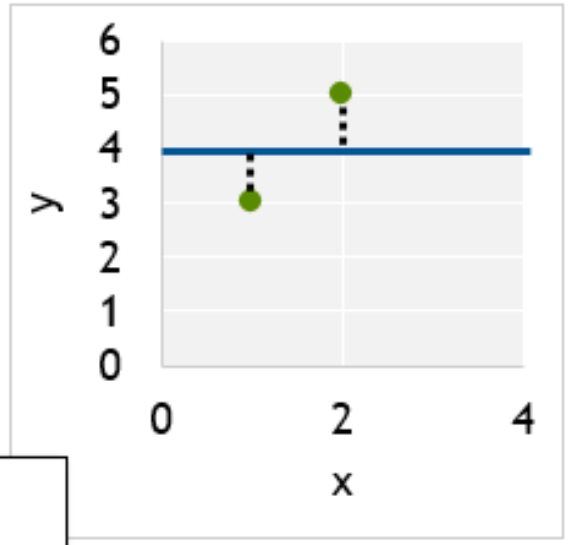


$m = -1$

$\mathrm{b} = 4$



Derin öğrenme

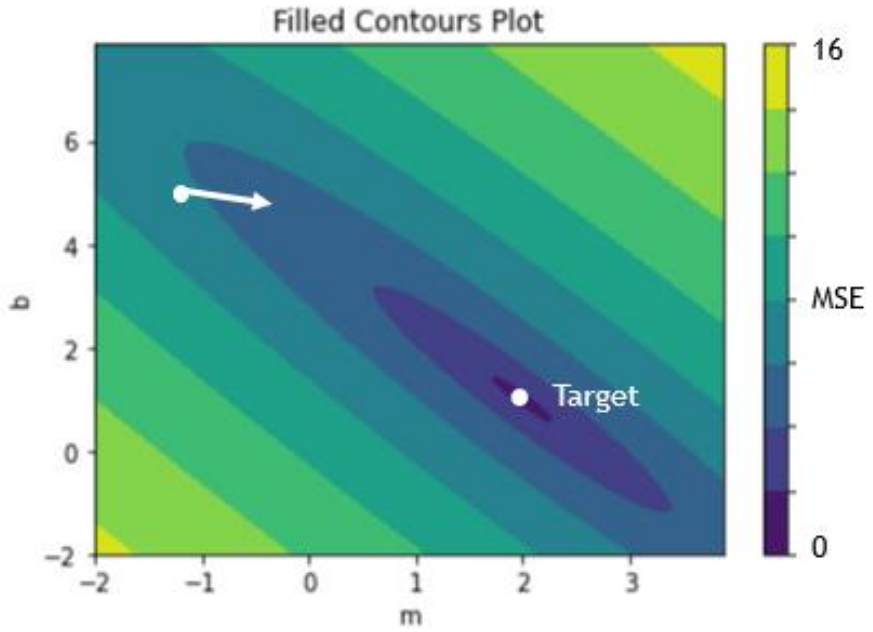


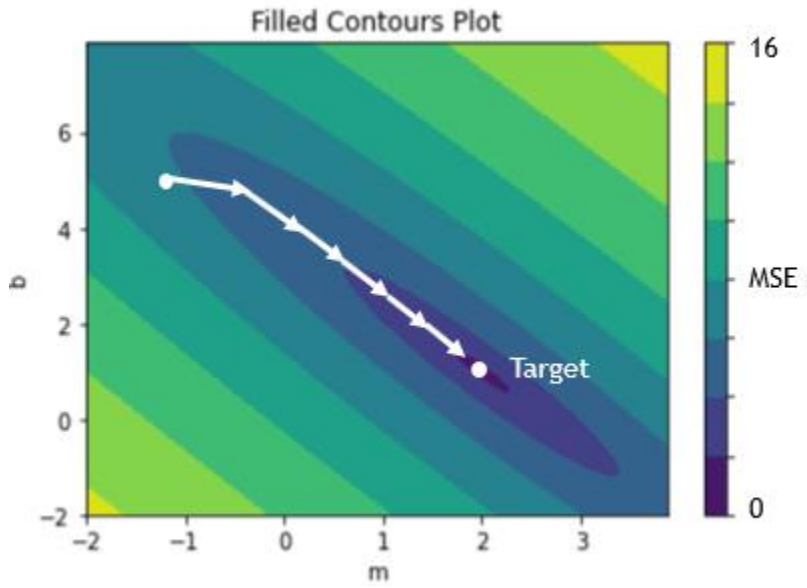
$m = 0$

$\mathrm{b} = 4$

Derin öğrenme

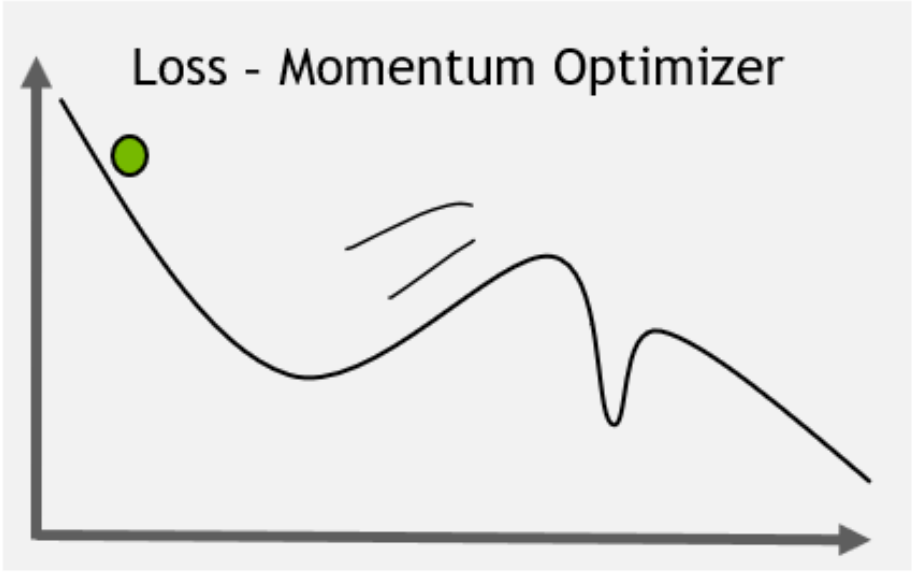
1. Gradyan: hangi yöne doğru hata en çok azalıyor
2. λ : öğrenme katsayısı
3. Epoch
4. Batch: örnek sayısı
5. Step: ağırlık parametrelerinin güncellenmesi





Derin öğrenme

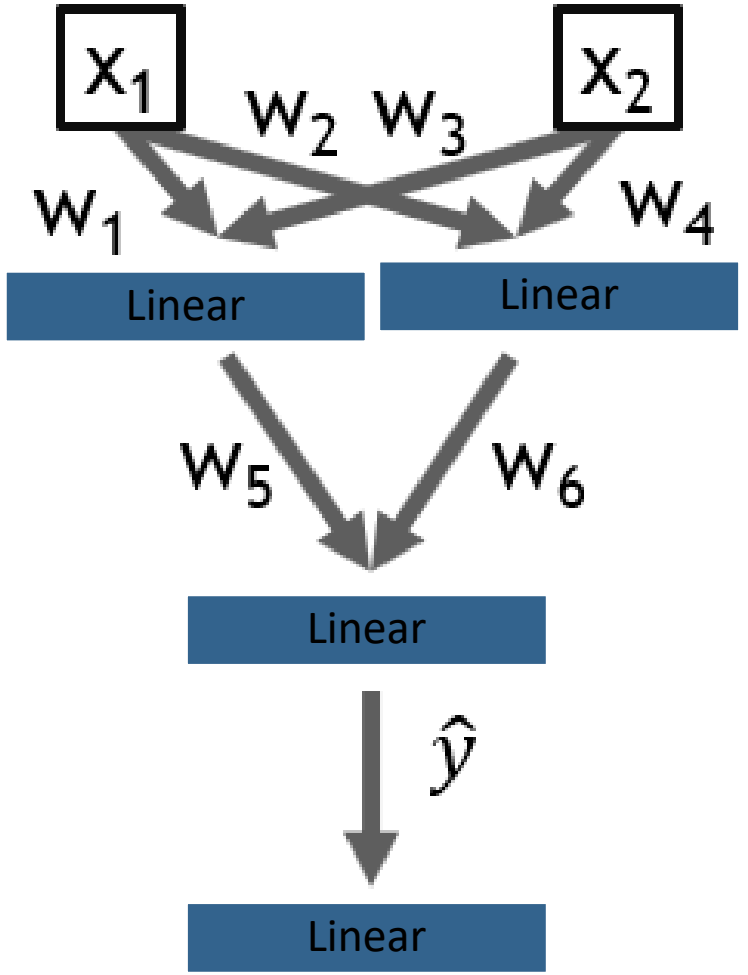
- SGD
- ADAM
- ADAGRAD
- RMSprop



Derin öğrenme

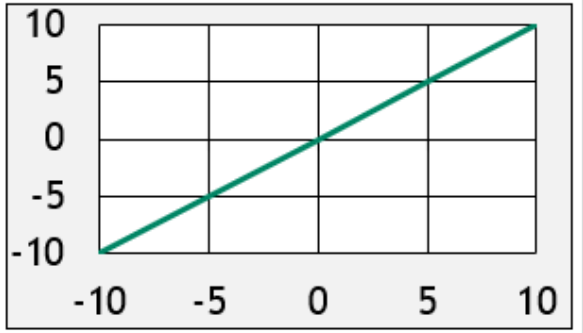
- Modelin oluşturulması

  - Scales to more inputs
  - Can chain neurons
  - If all regressions are linear, then output will also be a linear regression

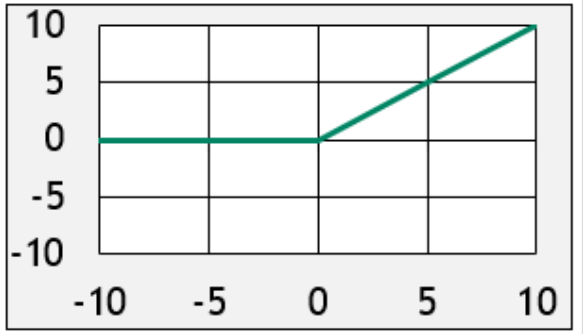


Derin öğrenme

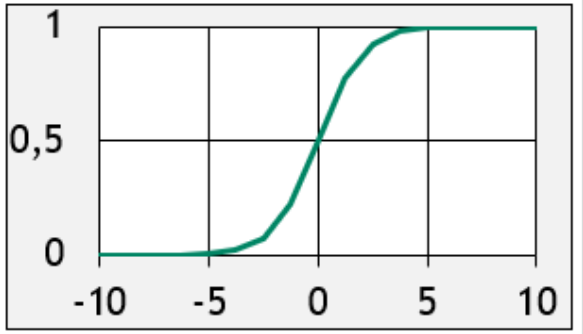
- Linear $\hat{y} = wx + b$



- ReLU $\hat{y} = \begin{cases} wx + b \; if \; wx + b > 0 \\ 0 \; otherwise \end{cases}$

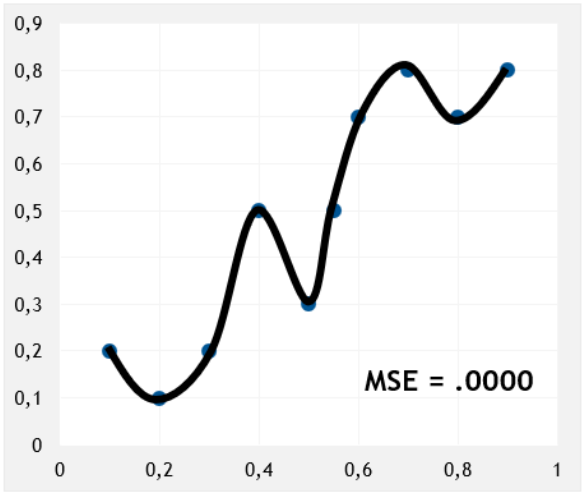


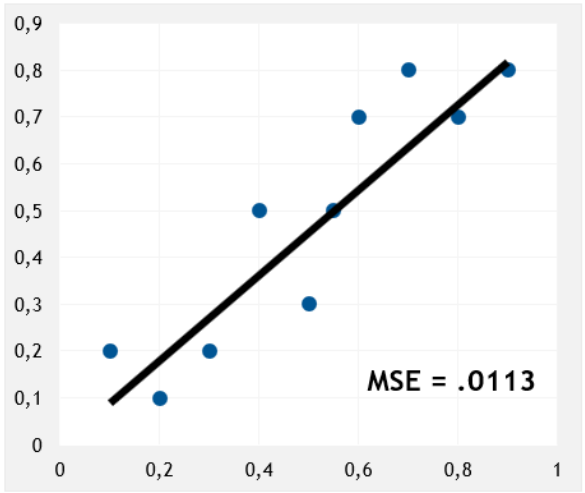
- Sigmoid $\hat{y} = \frac{1}{1 + e^{-(wx+b)}}$

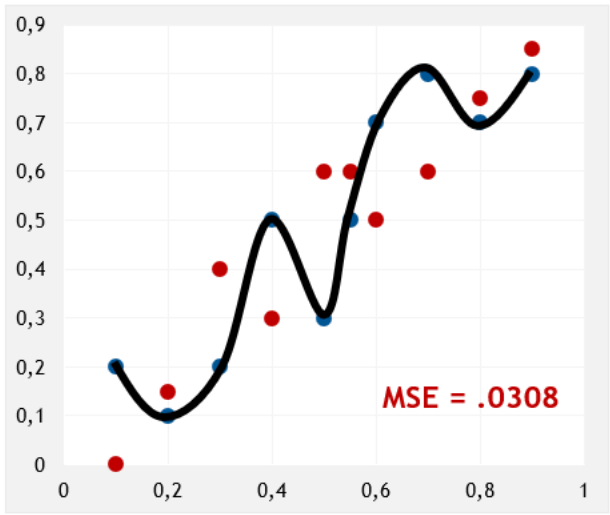


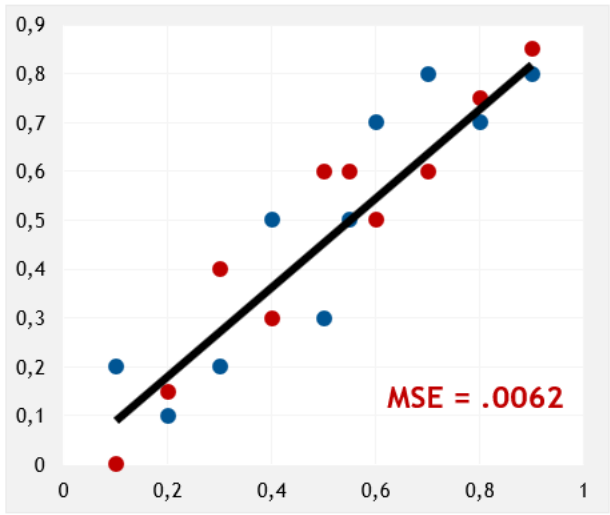
Derin öğrenme

- Overfitting



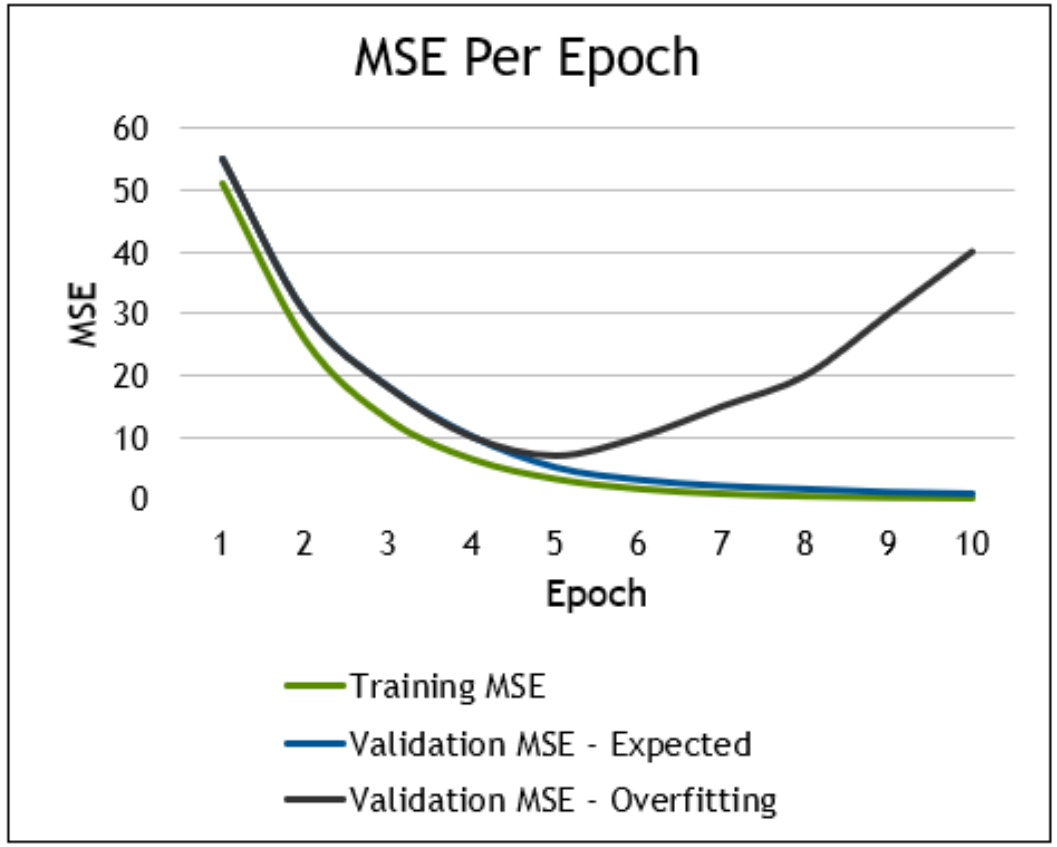




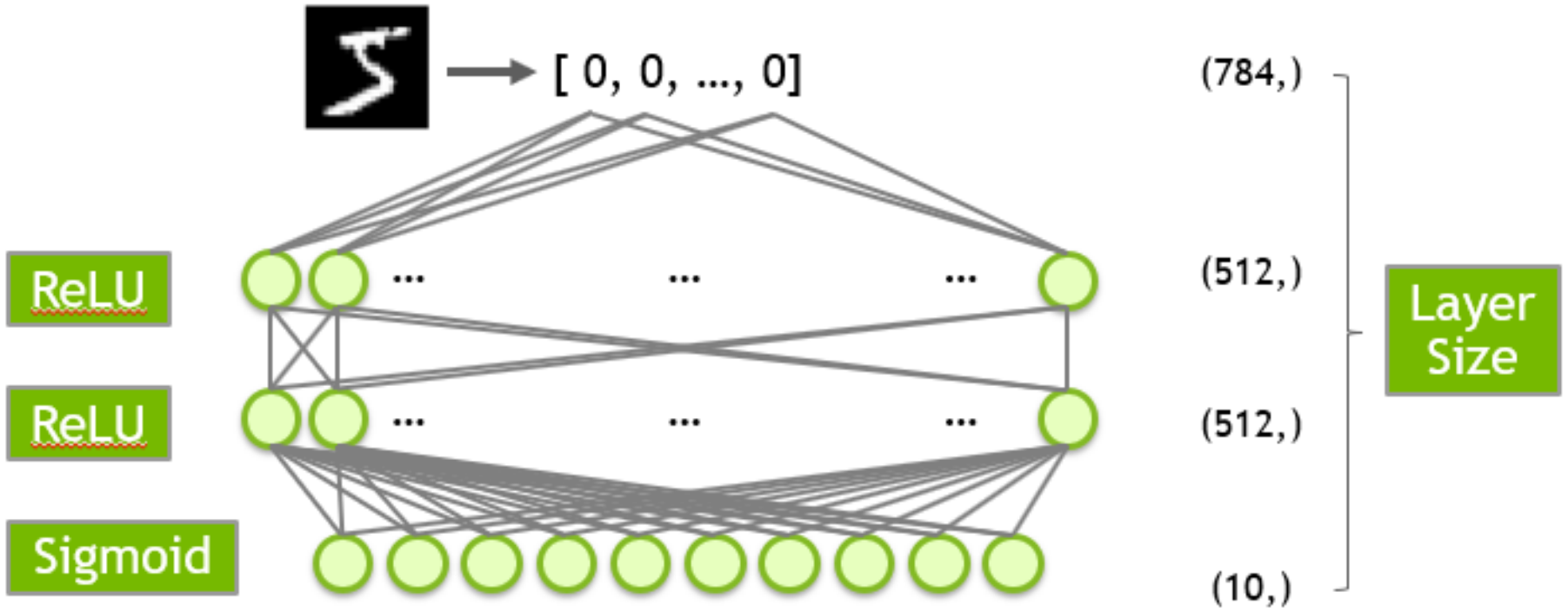


Derin öğrenme

- Training vs validation data
- Training: Core dataset for the model to learn on
- Validation: New data for model to see if it truly understands (can generalize)
- Overfitting: When model performs well on the training data, but not the validation data (evidence of memorization)
- Ideally the accuracy and loss should be similar between both datasets

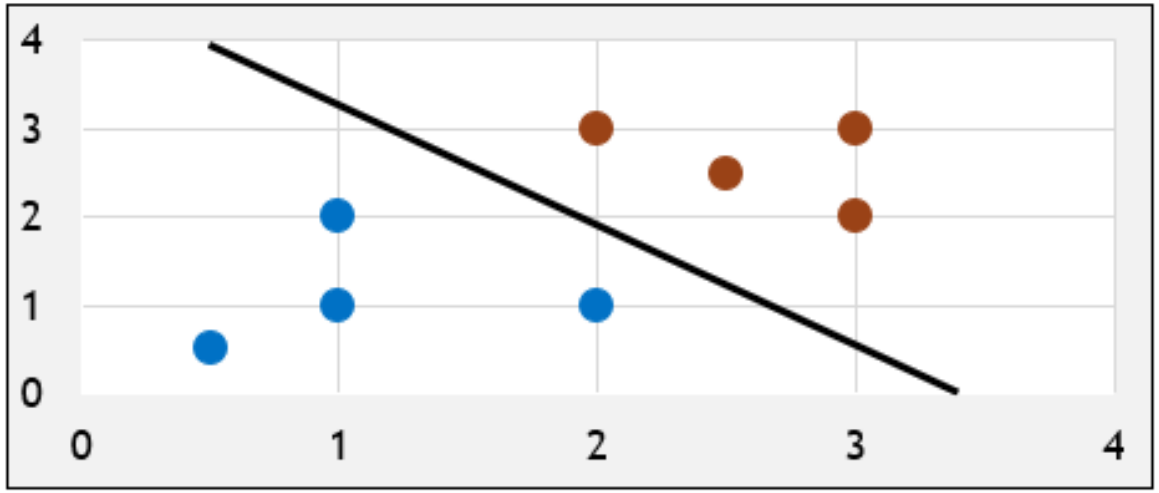


Derin öğrenme



Derin öğrenme

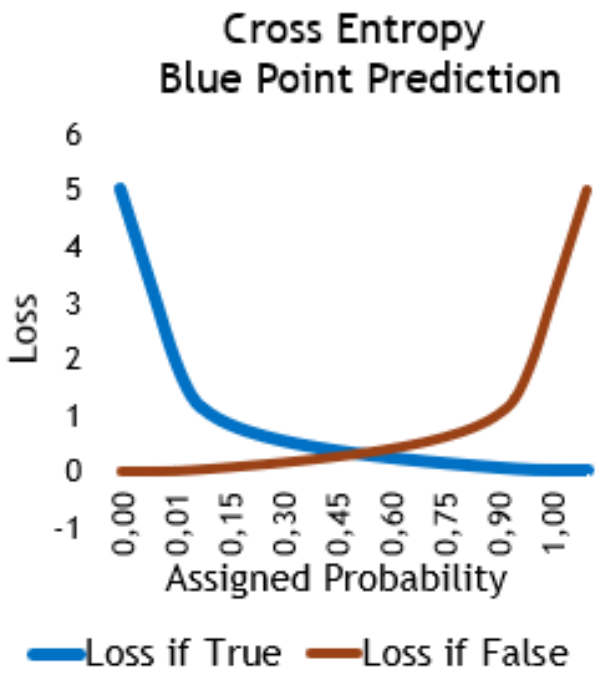
- Cross Entropy : *Etiketler ve tahminler arasındaki kaybı hesaplar.*



$$Loss = -\big((t(x) \cdot \log(p(x)) + \big(1 - t(x)\big) \cdot \log(1 - p(x))\big)$$

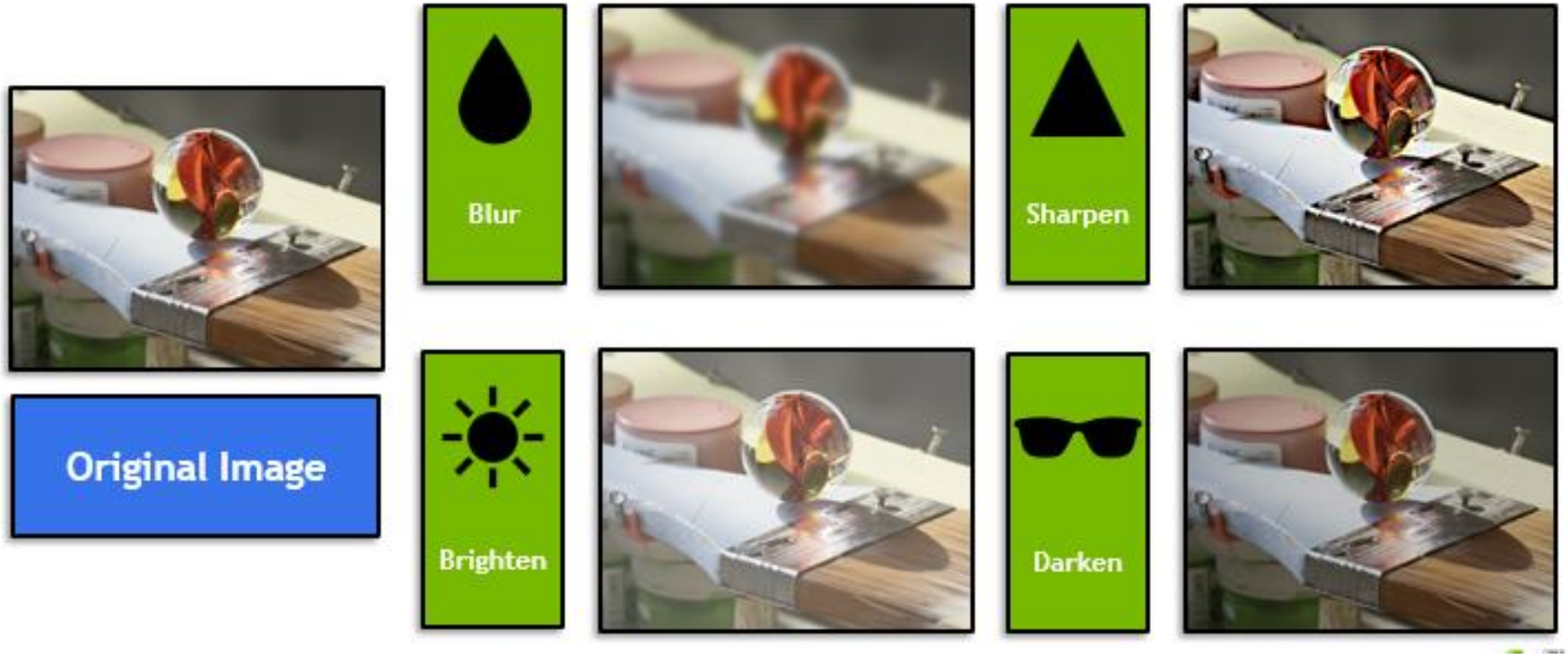
$$t(x) = target\ (0\ if\ False, 1\ if\ True)$$

$$p(x) = probability\ prediction\ of\ point\ x$$

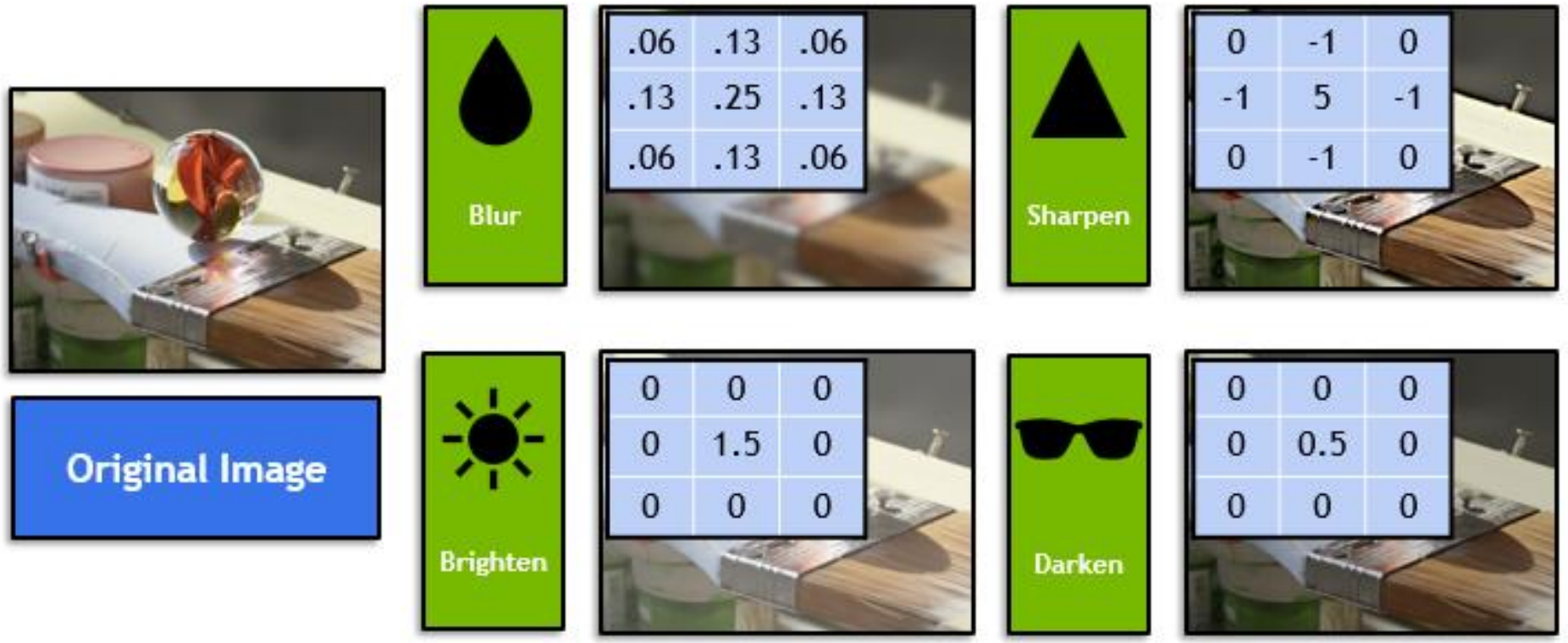


Derin öğrenme

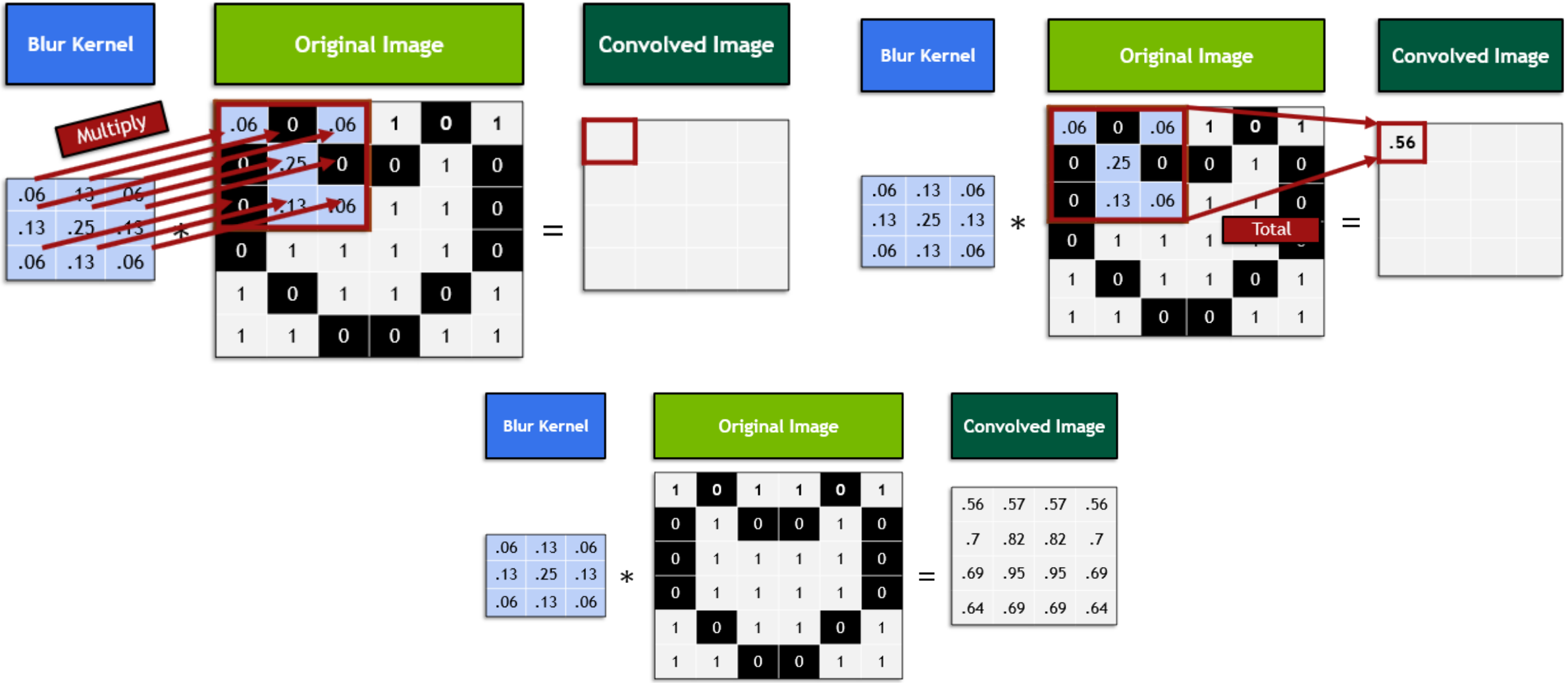
Kernels and Convolution



Derin öğrenme

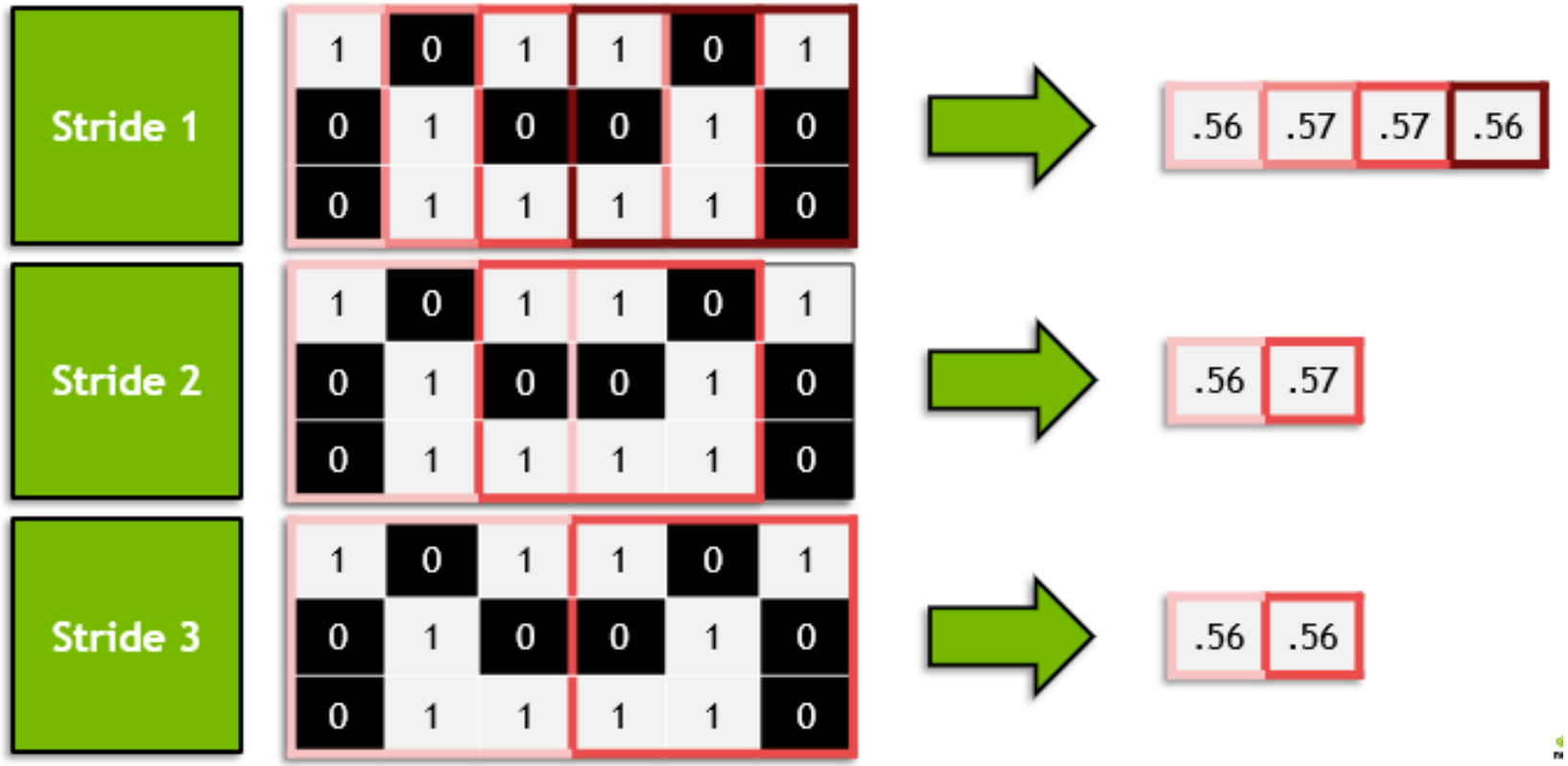


Derin öğrenme



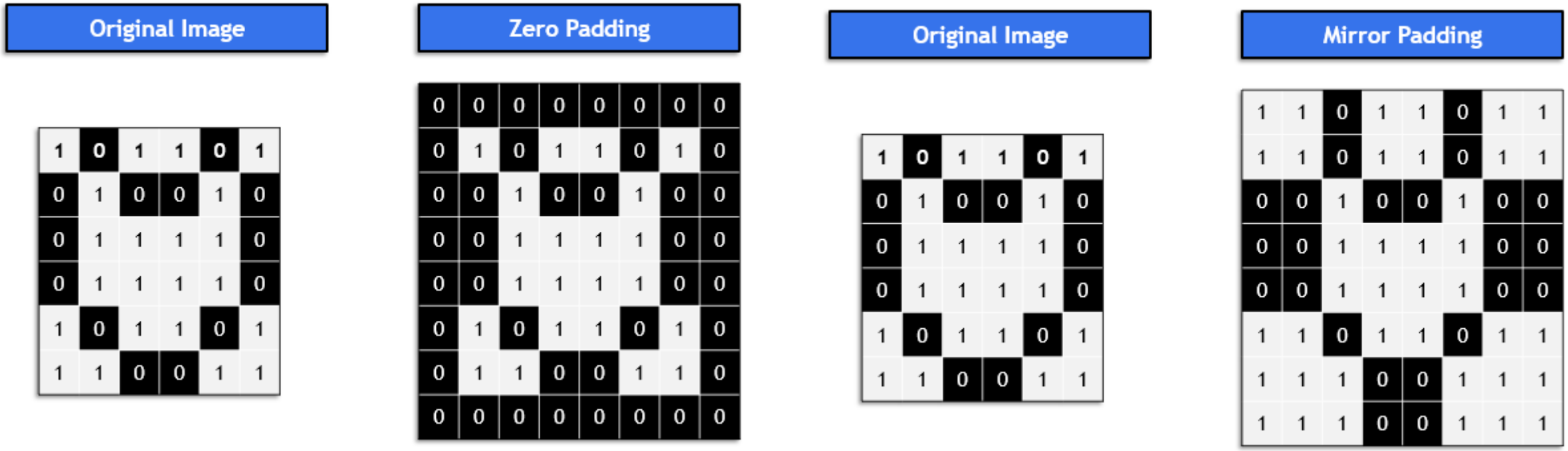
Derin öğrenme

- STRIDE



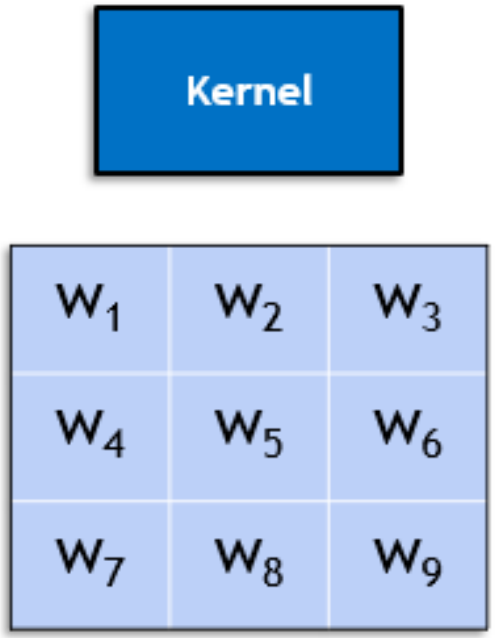
Derin öğrenme

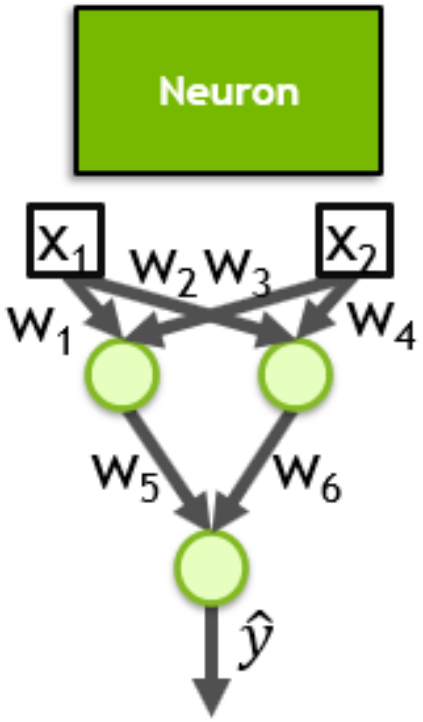
- Padding



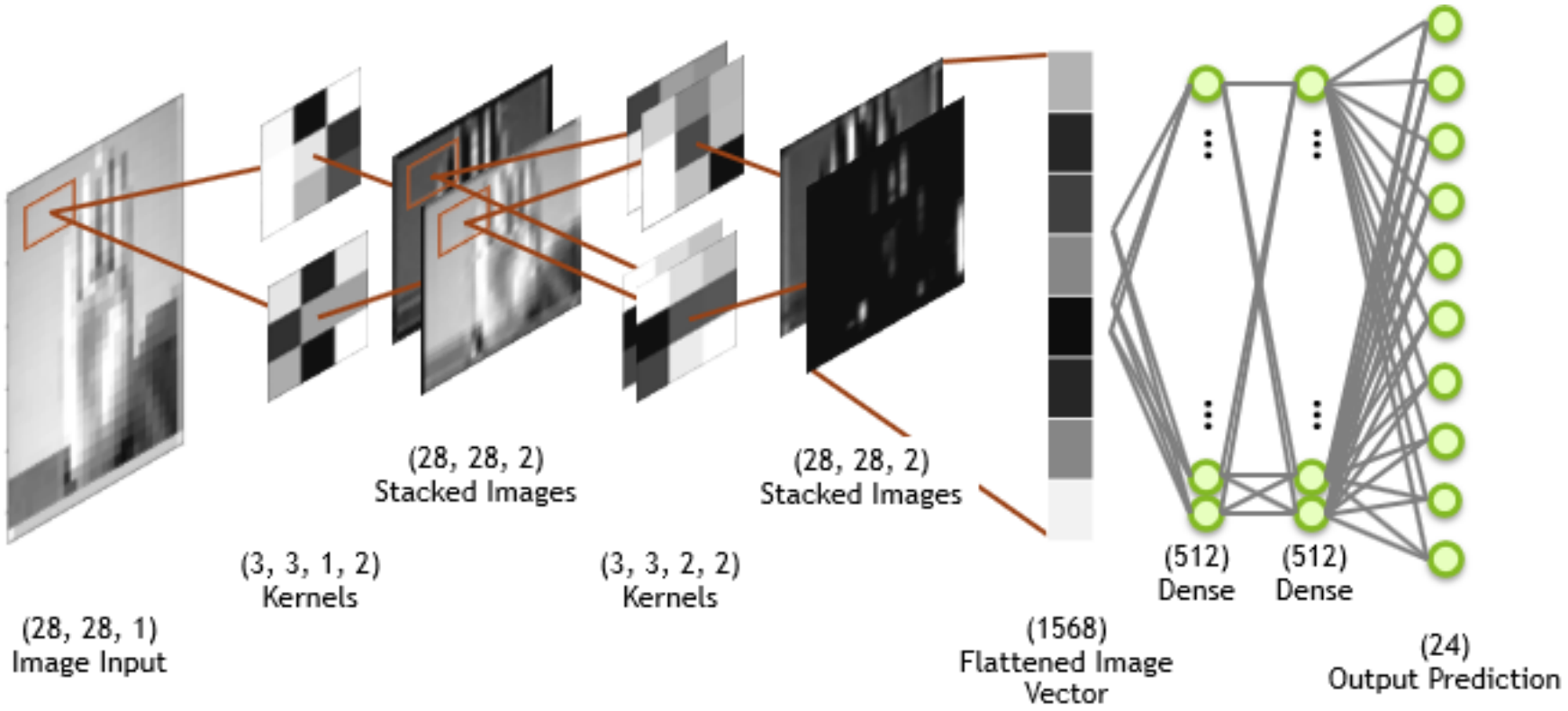
Derin öğrenme

KERNELS and Neural Networks



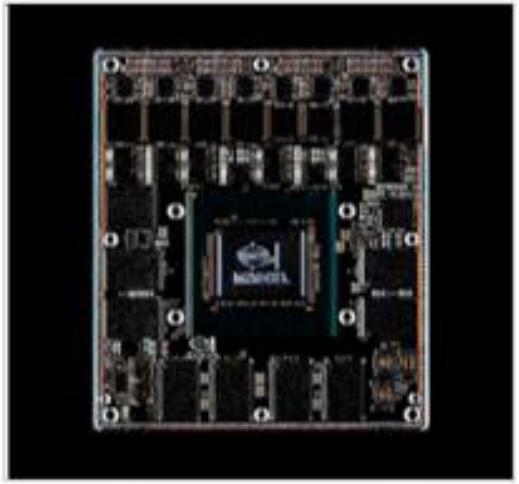


Derin öğrenme



Derin öğrenme

Vertical Edges

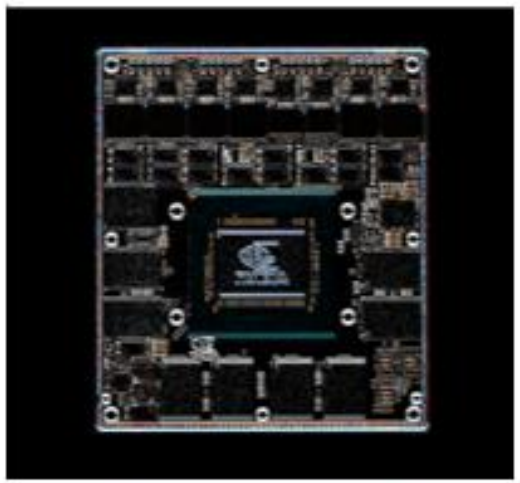


1	0	-1
2	0	-2
1	0	-1

Original Image

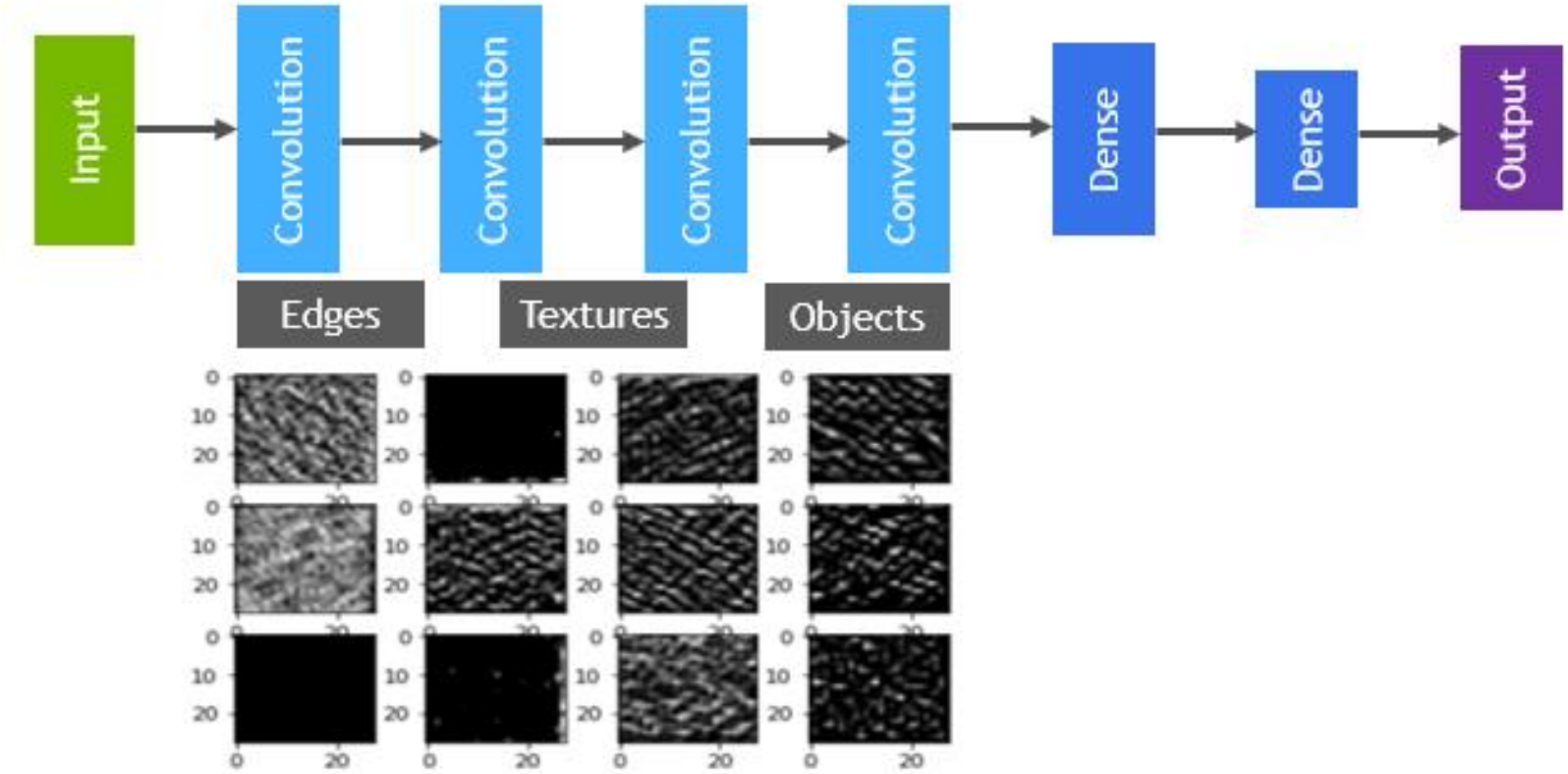
0	0	0
0	1	0
0	0	0

Horizontal Edges



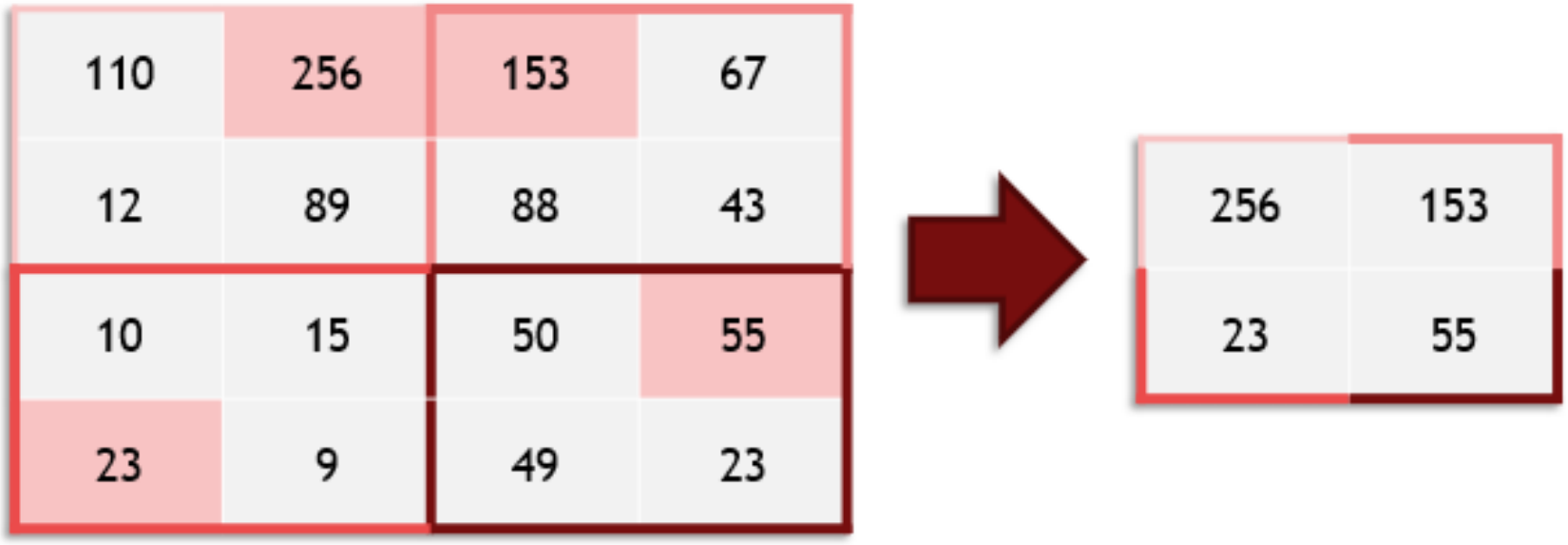
1	2	1
0	0	0
-1	-2	-1

Derin öğrenme



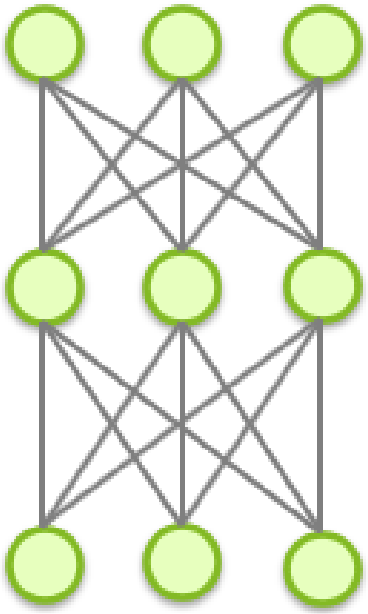
Derin öğrenme

Havuzlama Katmanı

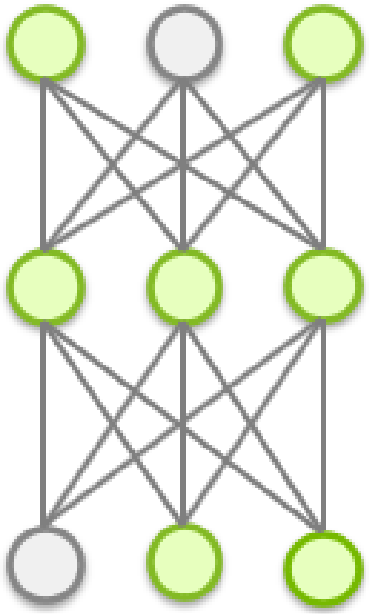


Derin öğrenme

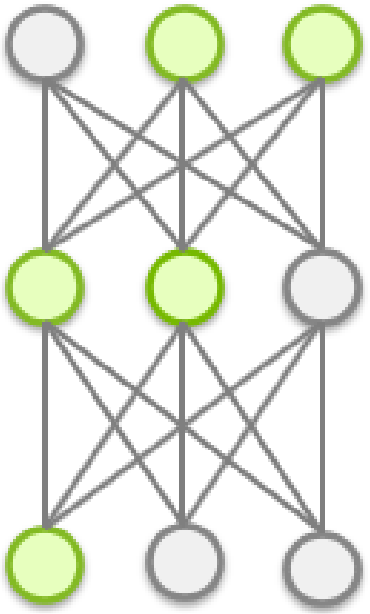
- Dropout



rate = 0



rate = .2



rate = .4