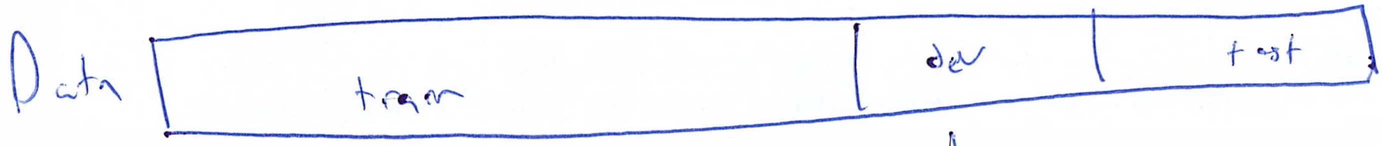
Kurs 2

Hafta 1



önce burda
hiperparametreler
olgunlaştır
sonra tamamını
uygula

ML ⇒ 70/30 yada 60/20/20

Big data ise ⇒ 1.000.000/10.000

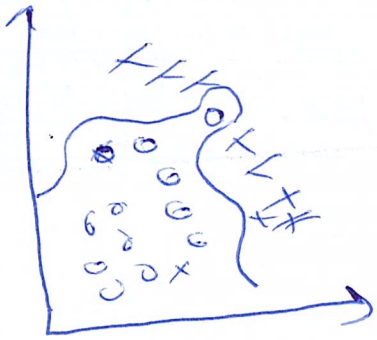
99/1 yada 98/1/1

bias: Jalik

Varyans: Varyans

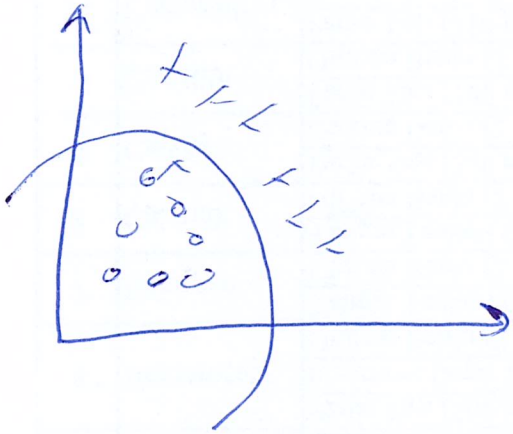
yüksek Jalik

high bias



yüksek varyans

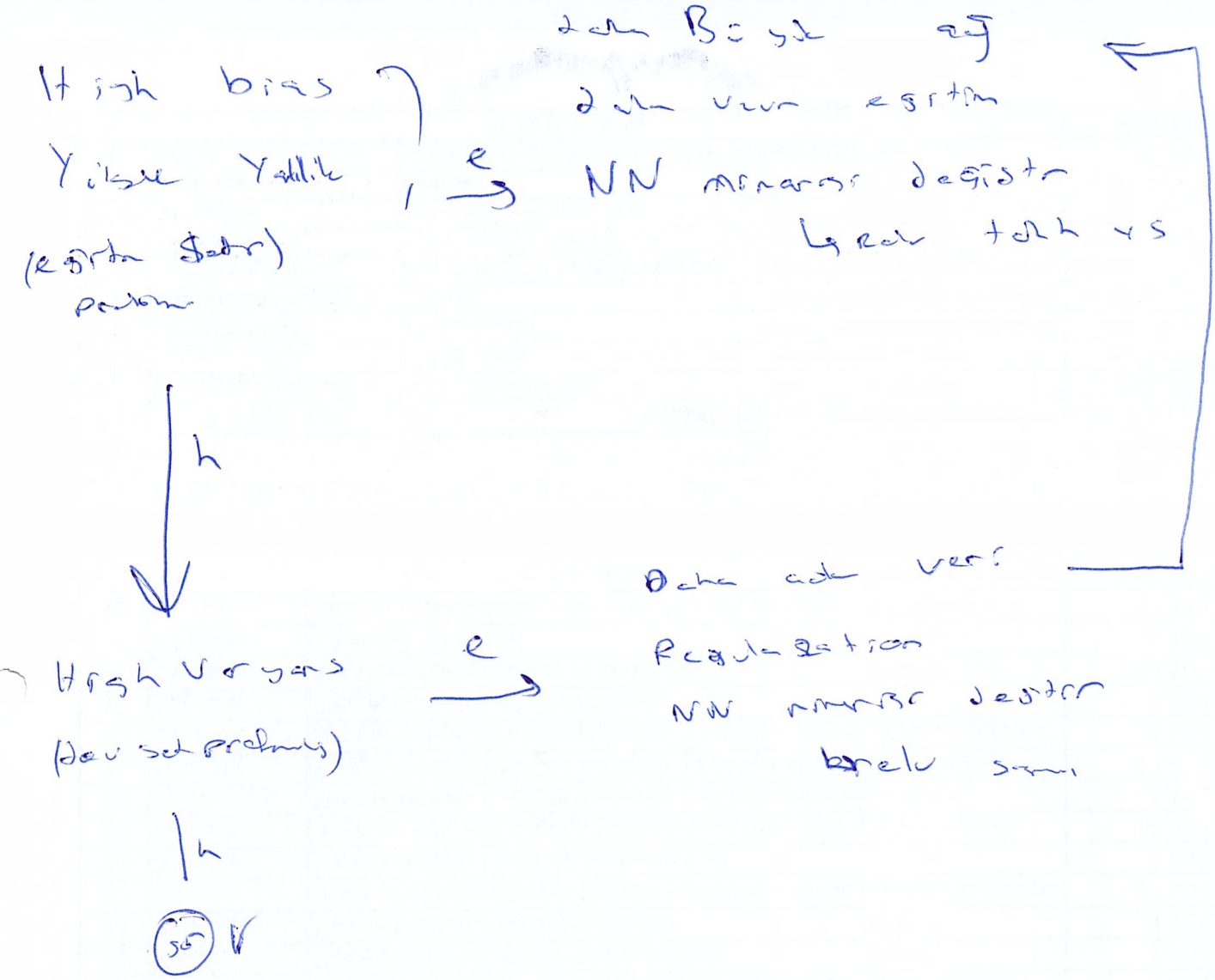
high variance



just right

q^{Bayes}				
train error	1%	15%	15%	0.5%
Dev error	11%	16%	30%	1%
	yüksek varyans	yüksek ~~Alak~~ bayılık	yüksek bayılık yüksek varyans	~~Yüksek~~ Düşük bayılık düşük varyans

varyans



Regularization ⟷ Düzenlileştirme

L2 Düzenlileştirme (daha yaygın)

$$J(w,b) = \frac{1}{m}\sum_{i=1}^{m} \mathcal{L}(\hat{y}^{(i)}, y^{(i)}) + \frac{\lambda}{2m}\|w\|_2^2$$

$$\|w\|_2^2 = \sum_{j=1}^{n_x} w_j^2 = w^T \cdot w$$

$$dw^{[l]} = \ldots + \frac{\lambda}{m} w^{[l]}$$

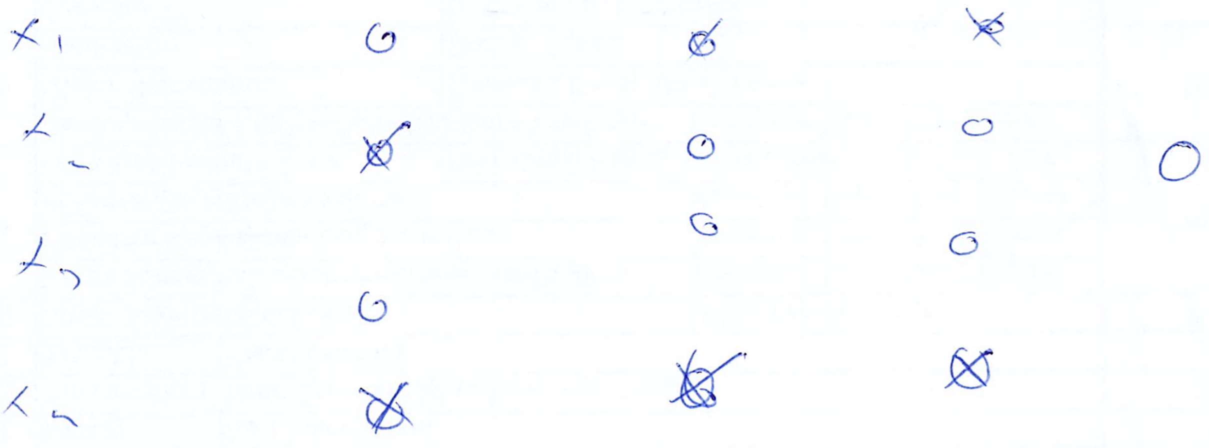
L1 regula

$$\frac{\lambda}{2m} \sum_{i=1}^{nx} |w| =$$

λ = düzenlileştirme parametresi

regularization parameter

Drop out düzenlileştirme 0.5 ⇒ %50



rastgele düğümleri sil

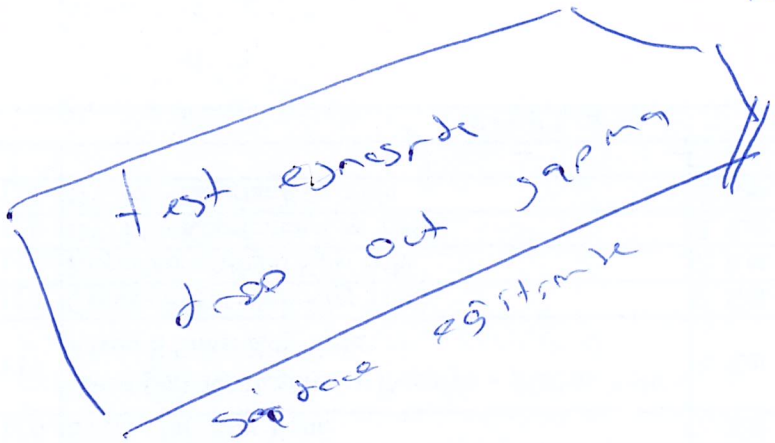
ancak kullanılan ters seyreltme

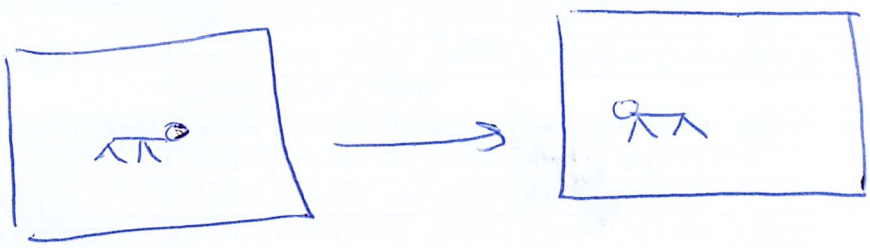
keepProb = 0.5
0.7
0.2
} silinecek nöronlar (0.2 ise %80'i)

d3 = np.random.rand(a3.shape) < keepProb

a3 = np.multiply(a3, d3) a3 *= d3

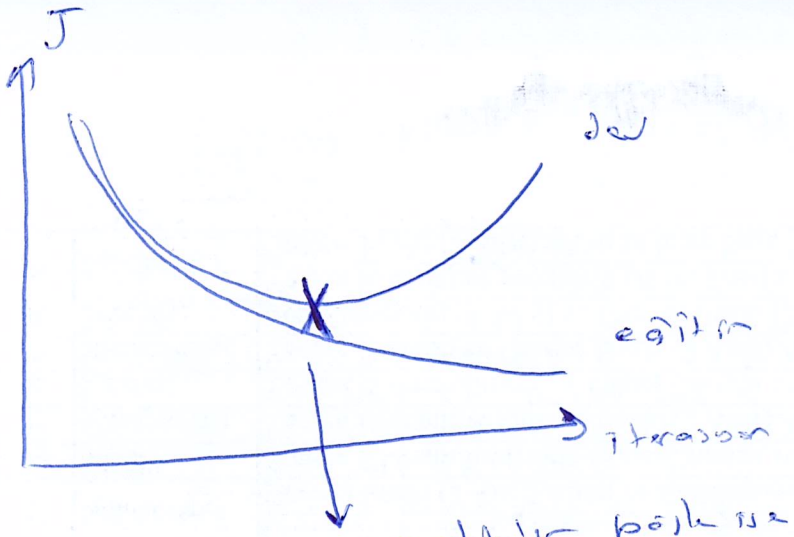
a3 = a3 / keepProb → sağdaki değer sayısı azaltılan oranda var olanı büyütür





data artırma

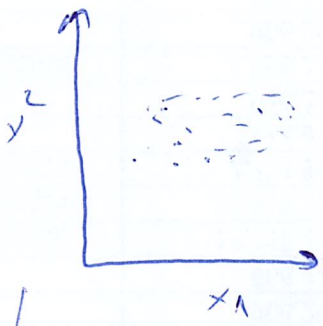
↳ zoom yap
rastgele kırp
çevirerek kırp



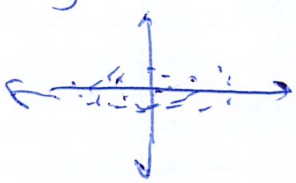
Normalize input

$$\mu = \frac{1}{m} \sum_{i=1}^{m} x^{(i)}$$

$$x = x - \mu$$

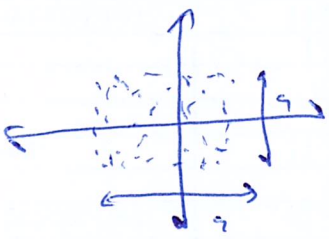


normalize



$$\sigma^2 = \frac{1}{m} \sum_{i=1}^{m} x^{(i)} **2$$

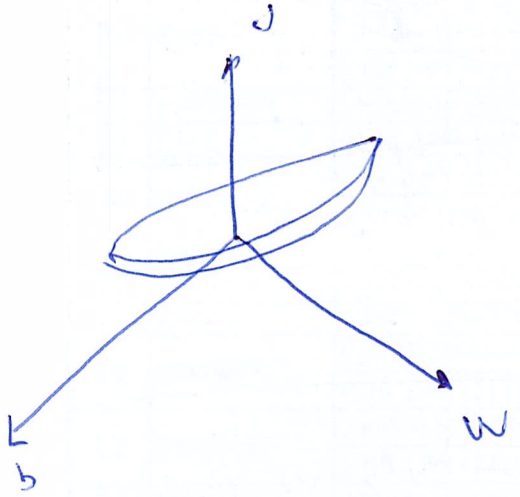
$$x = x / \sigma^2$$

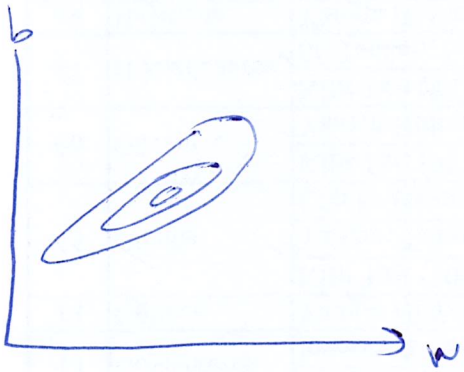


μ ve σ^2 eğitmeden test verisi için aynı olmalı

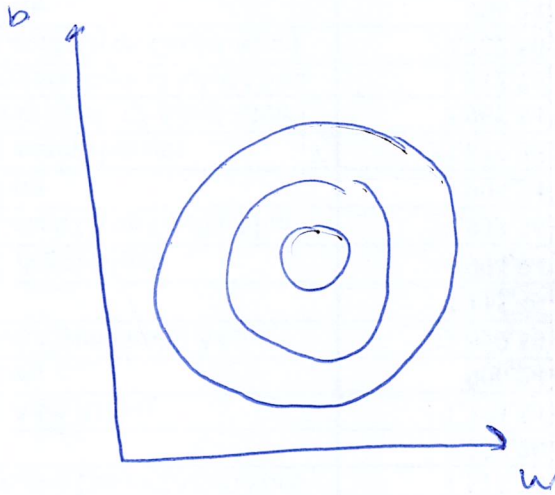
Normalisation

Once





Sonny



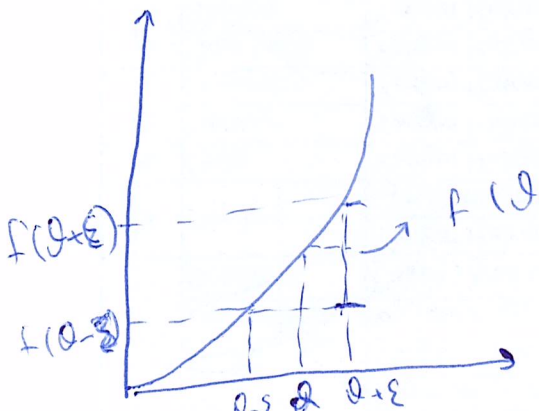
$$W^{[l]} = np.random.randn(shape) * np.sqrt\left(\frac{2}{n^{[l-1]}}\right)$$

ReLU, Tan
↓
sigmoid, Tan
↓
tanh

Xavier

$$\sqrt{\frac{2}{n^{(l-1)} \cdot n^{(l)}}}$$

$f(\theta) = \theta^3$



$f(\theta+\varepsilon) - f(\theta-\varepsilon)$

$\varepsilon = 0.01$

$$\frac{f(\theta+\varepsilon) - f(\theta-\varepsilon)}{2\varepsilon} \approx g(\theta)$$

$$g(\theta) = f'(\theta) = 3\theta^2 = 3$$ ---- yaklaşık eşit

$\Downarrow$

$\varepsilon = 0.001$

Gradient check

Sayıca değerden kullanır

$J(w^{[1]}, b^{[1]}, \ldots, w^{[l]}, b^{[l]}) = J(\theta)$

çok uzun yazılır

birleşir

$dw^{[1]}, db^{[1]}, \ldots \quad d\theta$

$J(\theta) = J(\theta_1, \theta_2 \dots \theta_n)$

for each i:

$$d\theta_{approx}^{[i]} = \frac{J(\theta_1, \theta_2, \dots \theta_i + \varepsilon, \dots) - J(\theta_1 \dots \theta_i - \varepsilon, \dots)}{2\varepsilon}$$

$$\approx d\theta^{[i]} = \frac{\partial J}{\partial \theta_i}$$

$$d\theta_{approx} \overset{?}{\approx} d\theta$$

Check $$\frac{\| d\theta_{approx} - d\theta \|_2}{\| d\theta_{approx} \|_2 + \| d\theta \|_2}$$

$\varepsilon = 10^{-7}$:dm

$\Rightarrow$

10^{-7} — great

10^{-5} — maybe ok

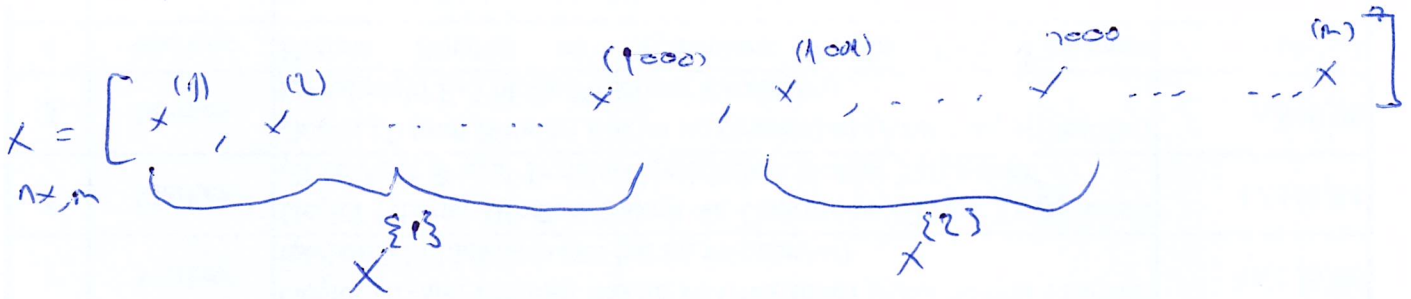
10^{-3} — worry

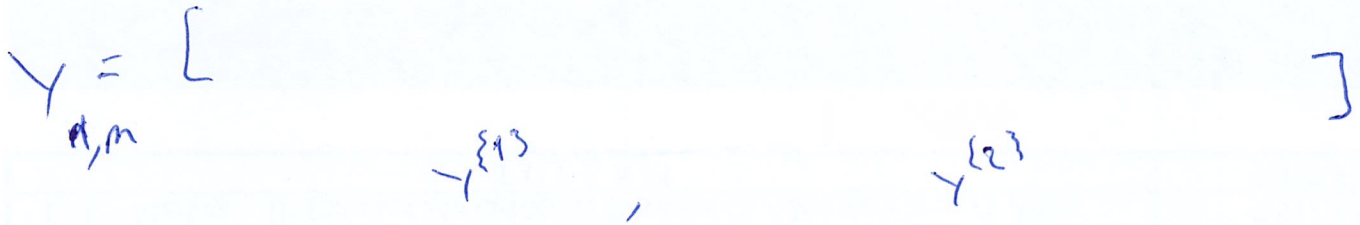
Mini batch

$m = 5000000$ olsa mesela

Parça parça alıcak her iterasyonda hızlı işlem seçiler.

mini batch boyutu 1000 olabilir mesela





mini batch gradyan inişi demek

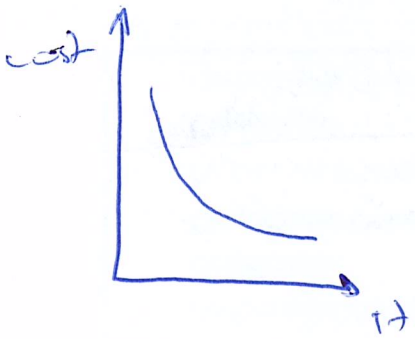
diğeri toplu " " idi

for i = 1 to 5000

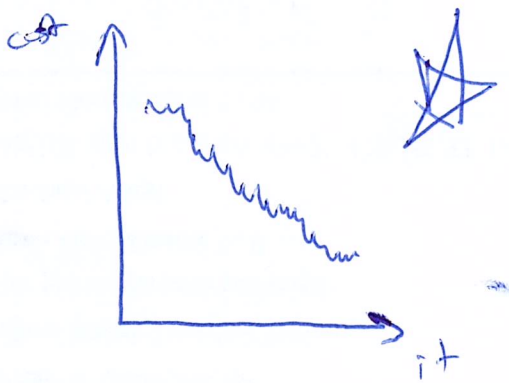
$X^{\{t\}}$ $Y^{\{t\}}$ kullan

her adım epoch eğitmek bir demek

Batch



mini batch

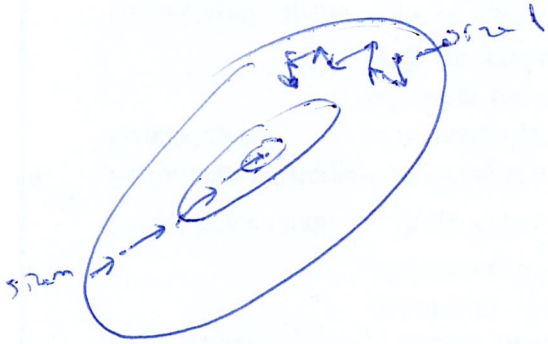


mini batch size = m

tüm eğim anlamında

if m = 1 Stochastik " "

⇒ 1 < m < M hızlı öğrenmedir



* mini batch size

yeni de 2^x şeklinde

olur (tanımlıdır)

en çok tercih edilen

değerler

$\approx \frac{1}{1-\beta}$ gün

~~A~~ Çok ince ekstra değerlendirme

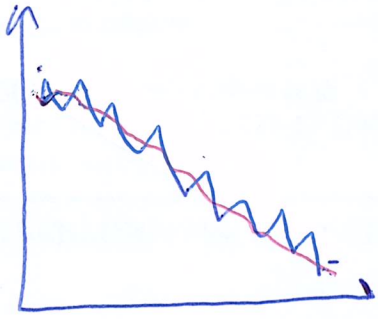
$\theta_1, \theta_2, \theta_3$ ... yeni değerler olsun

ortalamalar V_1, V_2 olsun

$$V_m = \beta V_{m-1} + (1-\beta)\theta_m$$

$\beta = 0.9 \Rightarrow$ 10 gün

$\beta = 0.98 \Rightarrow$ 50 gün



ilk şeyler için doğru çalışması için $V_0 = 0$ dır

bağlantısında bunu için

$$V_t = \frac{\beta V_{t-1} + (1-\beta) Q_t}{1 - \beta^t}$$

Momentum

β

$$V_{dw} = \beta V_{dw} + (1-\beta)\, dW$$

$$W = W - \alpha V_{dw}$$

RMS prop

β_2

$$S_{dw} = \beta_2 S_{dw} + (1-\beta_2)\, dw^2$$

$$S_{db} = \beta_2 S_{db} + (1-\beta_2)\, db^2$$

$\varepsilon = 10^{-7}$

$$W := W - \alpha \frac{\partial w}{\sqrt{S_{dw}} + \varepsilon}$$

$$b := b - \alpha \frac{\partial b}{\sqrt{S_{db}} + \varepsilon}$$

Adam optimization algorithm

$V_{dw} = 0, \quad S_{dw} = 0 \qquad V_{db} = 0 \quad S_{db} = 0$

$$V_{dw} = \beta_1 V_{dw} + (1-\beta_1)\, dw \qquad \leftarrow \text{momentum } \beta_1$$

$$S_{dw} = \beta_2 S_{dw} + (1-\beta_2)\, dw^2 \qquad \leftarrow \text{RMSprop } \beta_2$$

$$V_{dw}^{corrected} = V_{dw} / (1-\beta_1^t)$$

$$S_{dw}^{corrected} = S_{dw} / (1-\beta_2^t)$$

$$w = w - \alpha \frac{V_{dw}^{corrected}}{\sqrt{S_{dw}^{corrected}} + \varepsilon}$$

$\beta_1 = 0.9$

$\beta_2 = 0.999$

$\varepsilon = 10^{-8}$

} Gemäßle

Ogame error gradient

$$\alpha = \frac{1}{1 + \text{decay rate} * i} * \alpha_0$$

↓ epoc num

$\alpha_0 = 0.05$

decay rate = 1

$\alpha = 0.95^{(i)} \alpha_0$ — ex decay

$\alpha = \frac{k}{\sqrt{i}} \cdot \alpha_0$

Kurs 2

Hafta 3

hiper parametreler

α → en önemli

β → 2. önemli

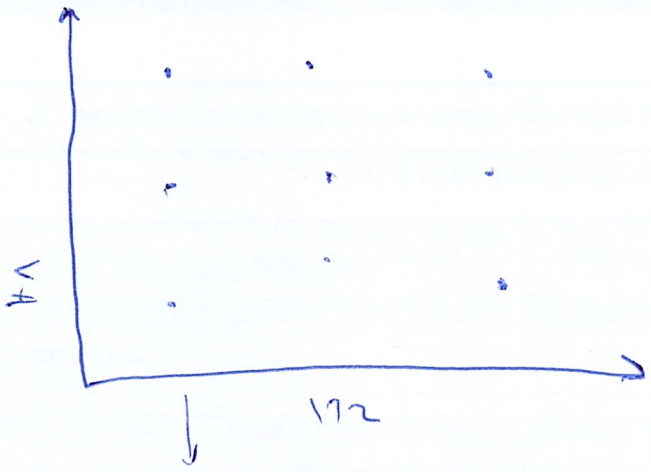
β_1, β_2, ε

layers

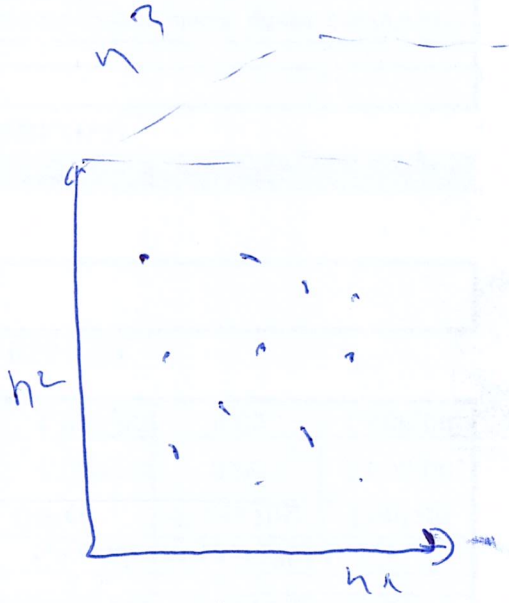
hidden units

learning rate decay

mini batch size



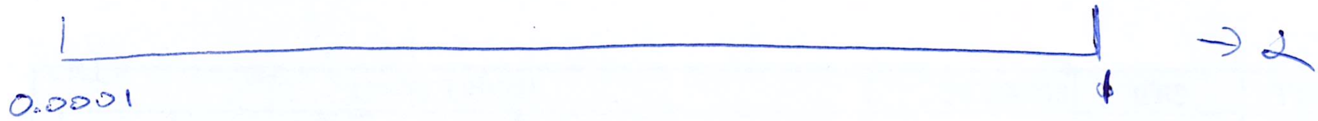
Burda sanki #1 önemli olsun



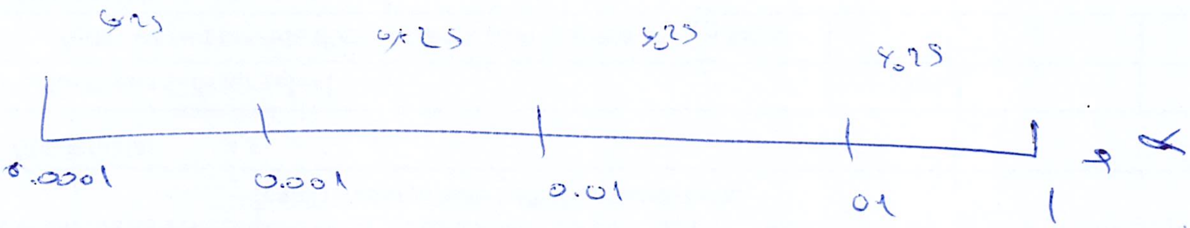
burda ise tane önemli olsun

daha sonra
575
var olur

burasa odaklan



0.0001 → 0.1 e kadar %10 olur. buyalış
0.1 → 1 " " %90



doğru

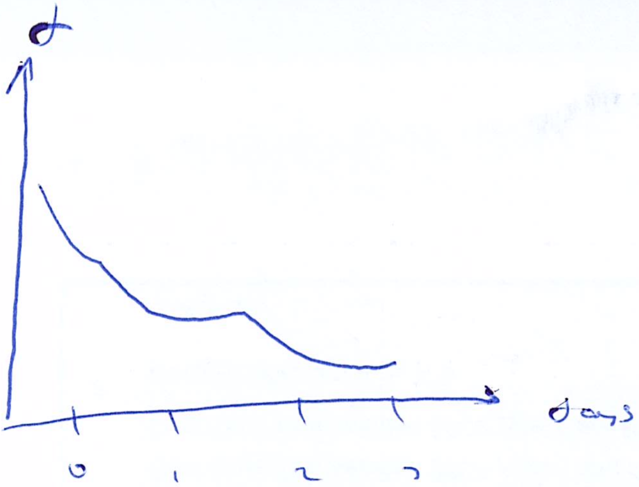
n = -4 * np.random.rand()

$\alpha = 10^{n}$

β 0.9 0.999 1 − B = 0.1 .. 0.001 => 10^{r}

↓ ↓ $\beta = 1 - 10^{r}$

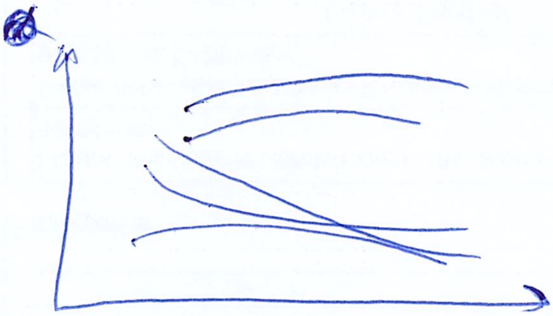
10 1000 adet



babysitting one model

Update α every day

Yada bor ede modelr asm ärle egrt



$$\mu = \frac{1}{m} \sum x^{(i)}$$

$$x = x - \mu$$

$$\sigma^2 = \frac{1}{m} \sum x^{(i)} ** 2$$

$$x = x / \sigma^2$$

$$\mu = \frac{1}{m} \sum_i z^{(i)}$$

$$\sigma^2 = \frac{1}{m} \sum_i (z_i - \mu)^2$$

$$z^{(i)}_{norm} = \frac{z^{(i)} - \mu}{\sqrt{\sigma^2 + \varepsilon}}$$

$$\tilde{z}^{(i)} = \gamma z^{(i)}_{norm} + \beta$$

learnable parameters

$\gamma = \sqrt{G^2 + \varepsilon}$

$\beta = \sqrt{\mu}$

$A^{[l]} \xrightarrow{W^{[l]}\, b^{[l]}} Z^{[l]} \xrightarrow{\beta^{[l]}\, \gamma^{[l]}} \tilde{Z}^{[l]} \rightarrow A^{[l+1]}$

batch norm

$g^{[l]}(\tilde{Z}^{[l]})$

$\mu = \frac{1}{m} \sum_i z^{(i)}$

$\sigma^2 = \frac{1}{m} \sum_i (z_i - \mu)^2$

$z^{(i)}_{norm} = \frac{z^{(i)} - \mu}{\sqrt{\sigma^2 + \varepsilon}}$

$\tilde{z}^{(i)} = \gamma z^{(i)}_{norm} + \beta$

Softmax

$X \rightarrow \cdots \rightarrow \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \rightarrow \hat{y}$

(4, 1)

1 cat
2 dog
3 chicken
4 degrl

$$z^{[L]} = w^{[L]} a^{[L-1]} + b^{[L]}$$

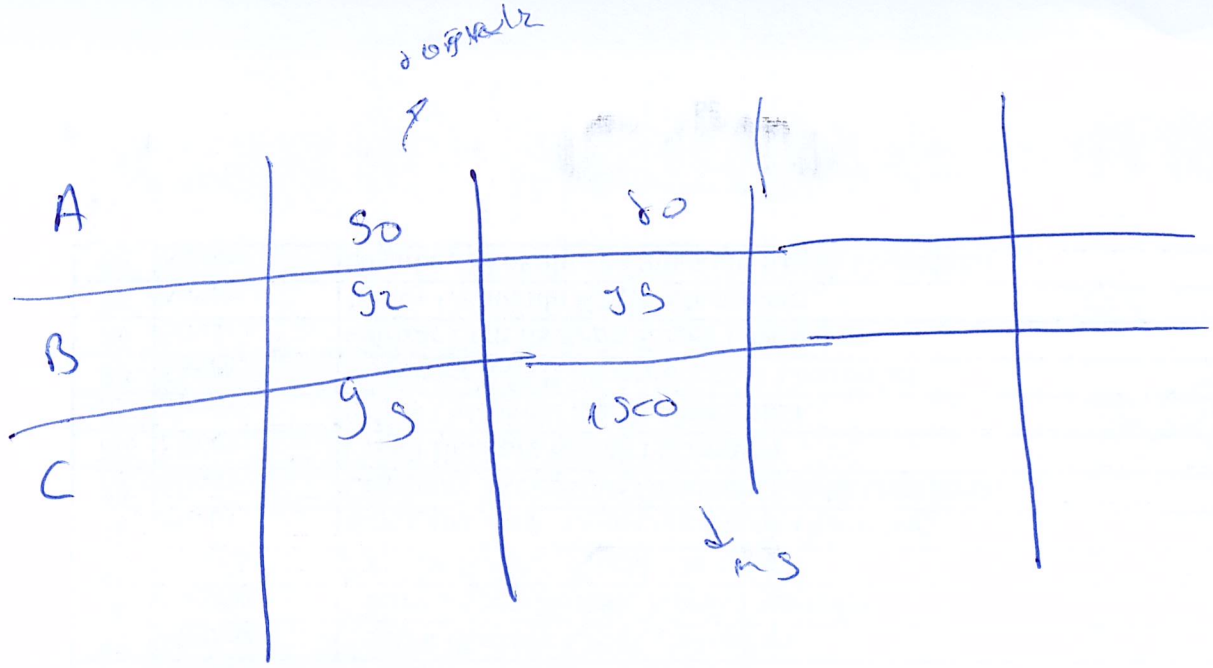
$$t = e^{(z^{[L]})}$$

$$a^{[L]} = \frac{e^{z^{[L]}}}{\sum_{j=1}^{4} t_i}$$

Softmax

$$z^{[L]} = \begin{bmatrix} 5 \\ 2 \\ -1 \\ 3 \end{bmatrix} \Rightarrow t = \begin{bmatrix} e^5 \\ e^2 \\ e^{-1} \\ e^3 \end{bmatrix} = \begin{bmatrix} 148.4 \\ 7.4 \\ 0.4 \\ 20.1 \end{bmatrix}$$

$$a^{[L]} = \begin{bmatrix} 0.842 \\ 0.042 \\ 0.002 \\ 0.114 \end{bmatrix}$$



cost = acc − 0.5 runs time

100 ms iyileşir

train set	dev set	test set
* eğitir	* farklı algoritmaları test edip en başarılı olanı bulmak için	* en son modelin performansını ölçmek için
* büyük		

Varyans ayrı başlıktan ele alınacak.