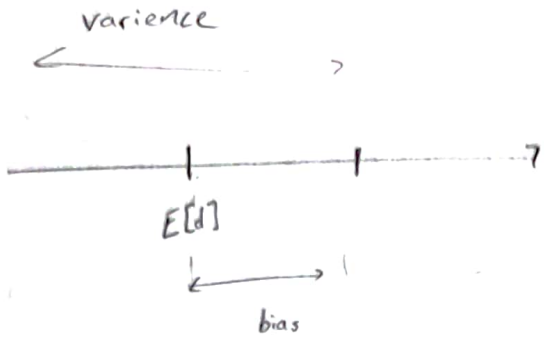
Bias and Varience



$$\text{Bias} = E[d] - \theta_1$$

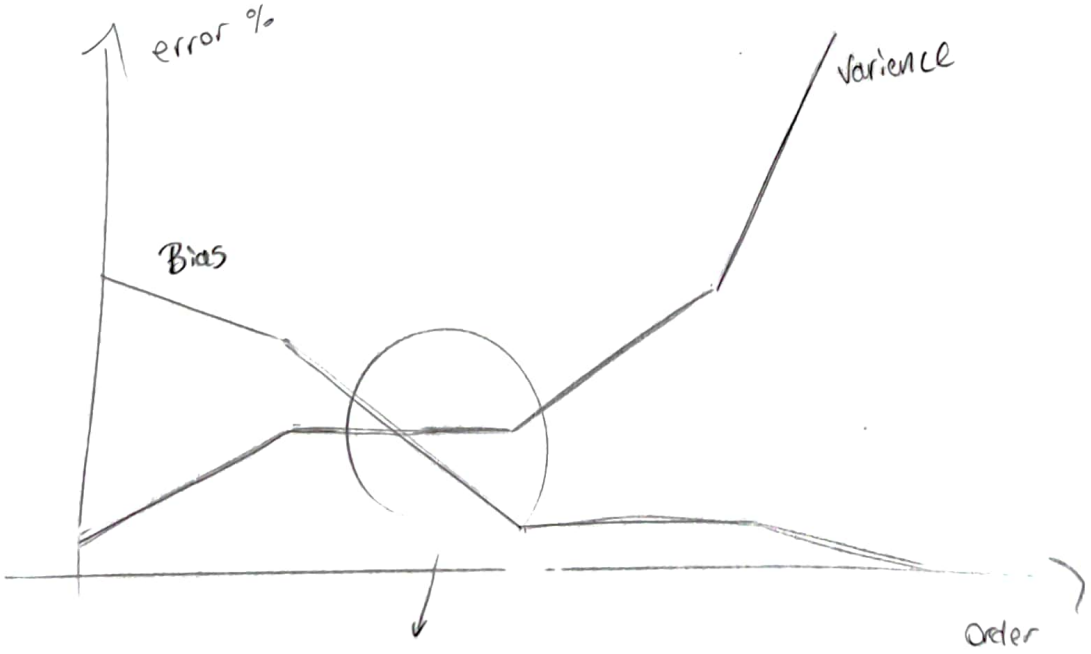
$$\text{Variance} = E\left[(d - E[d])^2\right]$$

mean square error

$$E\left[(d - \theta_1)^2\right] = \underbrace{\left(E[d] - \theta_1\right)^2}_{\text{bias}^2} + \underbrace{E\left[(d - E[d])^2\right]}_{\text{variance}}$$

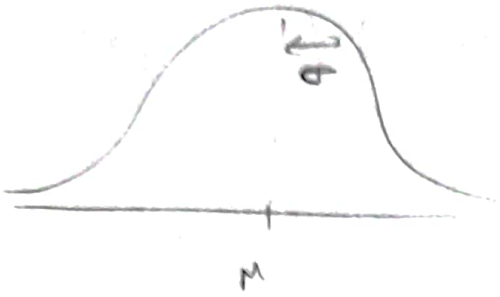
bias / variance dilemma

few parameters	lots of parameters
low variance	high variance
high bias	low bias



We use cross-validation to find this sweet spot.

Parameter Estimation



Gaussian (Normal) Distribution

$x^t \sim N(\mu, \sigma^2)$

$p(x) = N(\mu, \sigma^2)$

$$p(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{\left[-\frac{(x-\mu)^2}{2\sigma^2}\right]}$$

maximum likelihood estimate

↳ we will use it for classification

$$P(C/X) = \frac{P(X/C) \cdot P(C)}{P(X)}$$

likelihood, prior

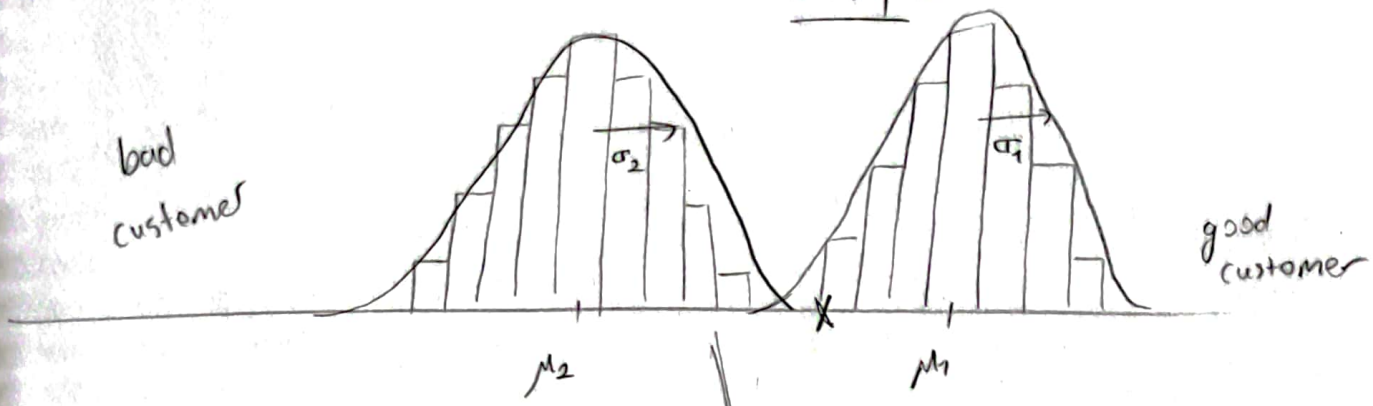
posterior prob. — evidence

Bayes again

$$P(C/X) \propto P(X/C)P(C)$$

proportional

classification example



$c=0$

$p(x \mid c=0)$

$\frac{\sum x^t}{N} = \mu_2$

$c=1$

$p(x \mid c=1)$

$\frac{\sum x^t}{N} = \mu_1$

$$p(x \mid c=0) = \frac{1}{\sqrt{2\pi\sigma_2^2}} \exp\left(-\frac{(x^t-\mu_2)^2}{2\sigma^2}\right)$$

$$p(x \mid c=1) = \frac{1}{\sqrt{2\pi\sigma_1^2}} \exp\left(-\frac{(x^t-\mu_1)^2}{2\sigma^2}\right)$$

$p(c_i \mid x) =$

$g_i(x) = \log P(x \mid C_i)$ $\quad P(c_i) = \log P(x \mid c_i) + \log P(C_i)$

$$g_i(x) = \log\left[\frac{1}{\sqrt{2\pi\sigma_i^2}} \cdot \exp\left(\frac{-(x-\mu_i)^2}{2\sigma^2}\right)\right] + \log P(C_i)$$

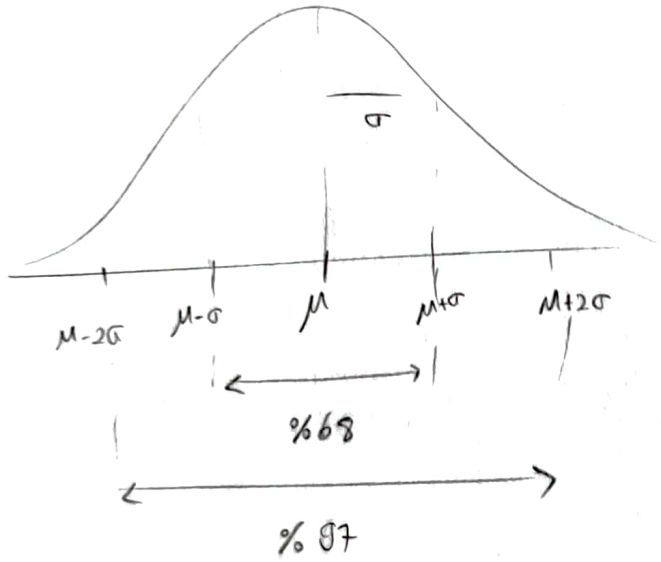
Error function = bias^2 + variance

	Bias	Variance
simple model	large	low
2nd	↓	↑
3rd		
i		

To find the minimum error; we use cross-validation.

a brief explanation ↑ of last lesson

Gaussian Dist.
(Normal)



$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \cdot \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

↓

it gives a density not probability.

To achieve probability we integrate for that point.

X_1	X_2	C
$x_1^{(1)}$	$x_2^{(1)}$	
$x_1^{(2)}$	$x_2^{(2)}$	
⋮	⋮	
$x_1^{(N)}$	$x_2^{(N)}$	

$$p(C=1 \mid x_1, x_2) = \frac{p(x_1, x_2 \mid C) \cdot p(C)}{p(x_1, x_2)}$$

$P(x_1, x_2) = P(x_1) \cdot P(x_2)$ (if they are independent)

$x_1 \sim N(\mu_1, \sigma_1^2)$

$x_2 \sim N(\mu_2, \sigma_2^2)$

$$P(x_1, x_2) = \left[\frac{1}{\sqrt{2\pi\sigma_1^2}} \cdot \exp\left(-\frac{1}{2} \cdot \frac{(x_1-\mu_1)^2}{\sigma_1^2}\right)\right] \cdot \left[\frac{1}{\sqrt{2\pi\sigma_2^2}} \cdot \exp\left(-\frac{1}{2} \cdot \frac{(x_2-\mu_2)^2}{\sigma_2^2}\right)\right]$$

$$\frac{1}{2\pi\sigma_1\sigma_2} \cdot \exp\left[\frac{1}{4} \cdot \frac{(x_1-\mu_1)^2}{\sigma_1^2} \cdot \frac{(x_2-\mu_2)^2}{\sigma_2^2}\right]$$

$$\begin{bmatrix} x_1-\mu_1 & x_2-\mu_2 \end{bmatrix} \begin{bmatrix} x_1-\mu_1 \\ x_2-\mu_2 \end{bmatrix}$$

$$X = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \qquad \mu = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}$$

$$(x-\mu)^T \cdot (x-\mu)$$

$$\begin{bmatrix} x_1-\mu_1 & x_2-\mu_2 \end{bmatrix} \begin{bmatrix} \frac{1}{\sigma_1^2} & 0 \\ 0 & \frac{1}{\sigma_2^2} \end{bmatrix} \begin{bmatrix} x_1-\mu_1 \\ x_2-\mu_2 \end{bmatrix}$$

$$\Sigma^{-1}$$

covariance matri

$$\Sigma = \begin{bmatrix} \sigma_1^2 & 0 \\ 0 & \sigma_2^2 \end{bmatrix}$$

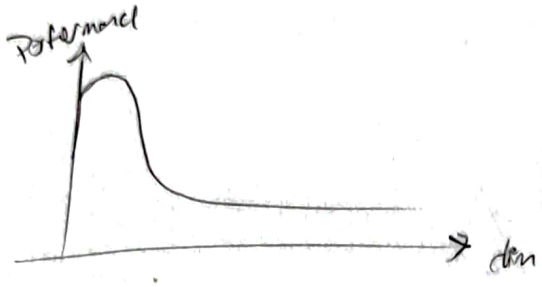
$$\Sigma = \Sigma^T \text{ (symmetric)}$$

* Dimensionality Reduction → reduces
  - time complexity
  - space complexity
  - cost of observing feature

↓
data visualization

curse of dimentionality

there is a point on number of features that the bigger we go above that point, performance starts to degrading instead of improving.



feature selection vs extraction

class labels

Selection → more correlation with class labels && less redundancy

* minimum redundancy
* maximum relevant

mrMR algorithm

extraction → project the original d to new k dim

k < d

* Principal component Analysis

PCA

(unsupervised model)

↓

NO class labels

eigen matrislerinden

$A_{d \times d}$ $\quad Ax = \lambda x$

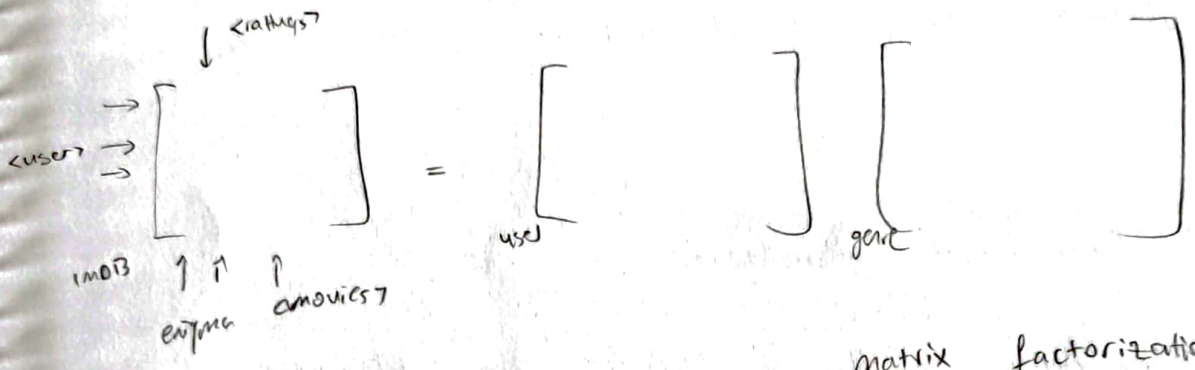
$(A - \lambda) x = 0$

$\det(\lambda I - A) = 0$

$$A \begin{bmatrix} | & | & & | \\ x_1 & x_2 & \cdots & x_n \\ | & | & & | \end{bmatrix} = \begin{bmatrix} | & | & & | \\ x_1 & x_2 & \cdots & x_n \\ | & | & & | \end{bmatrix} \cdot \begin{bmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_n \end{bmatrix}$$

$X \qquad\qquad X \qquad\qquad \Lambda$

$A = X \Lambda X^{-1}$ (if X^{-1} exists)



matrix factorization

* singular value decomposition

$A_{N \times d} = U \Sigma V^T$ $\quad$ Σ diagonal $\quad$ $U \perp V$ $\quad$ $U \cdot U^T = I$ $\quad$ $V \cdot V^T = I$

$A^T \cdot A = (U \Sigma V^T)^T U \Sigma V^T = \ldots = V \Sigma^2 V^{-1}$

$\sim V \Lambda V^{-1}$

$A^T \cdot A = B = V \Sigma^2 V^{-1}$

$A \cdot A^T = C = U \Sigma^2 U^{-1}$

Subset selection

- forward selection

  add the best feature at each step until it stops getting better

$f_1 \longrightarrow f_1, f_4 \longrightarrow f_1, f_4, f_7 \longrightarrow \cdots$

$O(d^2)$

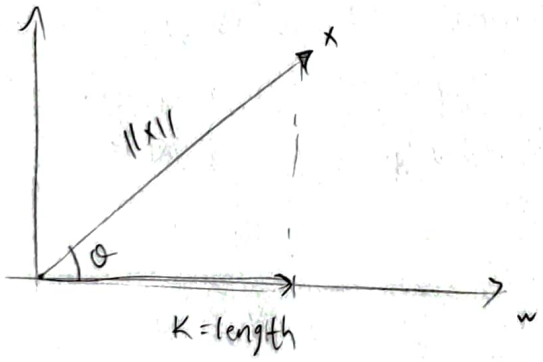
- backward selection

  same as forward but reverse

$f_1, f_2, \cdots f_d \longrightarrow f_1, f_2, f_5 \cdots f_d \longrightarrow$

Principal Component Analysis

Project the data on to z where the variance is maximized.



$\cos\theta = \frac{K}{\|x\|}$

$\downarrow$

$x^T w = \|x\| \cdot \|w\| \cdot \cos\theta$

$w^T x = \|x\| \cdot \|w\| \cdot \frac{K}{\|x\|}$

$K = \frac{w^T \cdot x}{\|w\|} \rightsquigarrow 1$

$K = w^T \cdot x$

after projection: $z = w^T \cdot x$

$\|w\| = 1$ find the w value that maximizes z.

$$\arg\max \; E\left[(w^T x - w^T \mu)^2\right]$$

$$= (w^T x - w^T \mu)\cdot(w^T x - w^T \mu) = (w^T x - w^T \mu)(x^T w - \mu^T w)$$

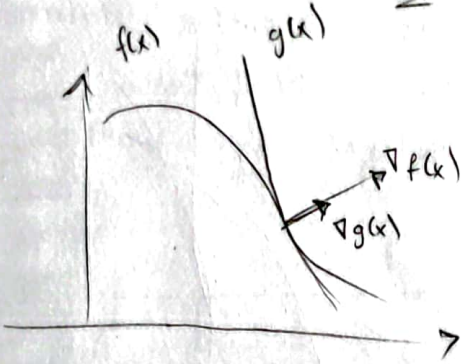
$$= w^T (x-\mu)(x^T - \mu^T) w = E\left[w^T (x-\mu)(x-\mu)^T w\right]$$

$$w^T \cdot E\left[(x-\mu)(x-\mu)^T\right] w$$

$\Sigma \rightarrow$ covariance matrix

subject to $\|w\| = 1$

constraint optimization



Both functions have the gradients that have same direction.

Lagrange Multipliers

$$\nabla f(x) = \lambda \nabla g(x)$$

$$\nabla f(x) - \lambda \nabla g(x) = 0$$

check out: derivative of vector (partial)
(least square

$$\frac{\partial L(w,\lambda)}{\partial w} = 0$$

$$\Sigma w = \lambda w$$

Σ: covariance matrix

λ: largest eigenvalue

w: corresponding eigenvector

$[X]_{N \times d}$ $\quad \Sigma_{d \times d}$

d eigen values

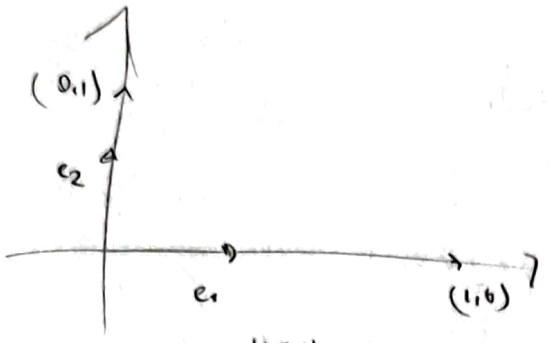
$d_1 \geq d_2 \geq d_3 \geq \cdots \geq d_d$

$w_1 \; w_2 \; \cdots \; w_n$

$[\;][\;]$ $\quad d \times 1$

$Z_1 = X \cdot w_1 \quad \rightsquigarrow \quad Z_k = X \cdot w_k$

* PCA

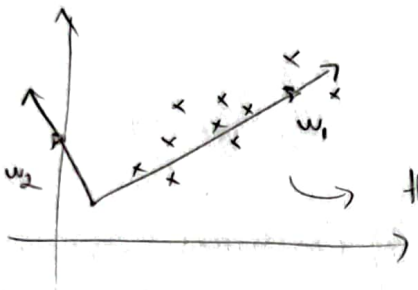


$\|e_1\| = 1$

$\|e_2\| = 1$

$\begin{bmatrix} 5 \\ 4 \end{bmatrix} = 5e_1 + 4e_2$

e_1 and e_2 are bases of dimensions



→ these eigenvectors can be our new bases so that we can visualize data better and that would reduce dimensions to whatever we want.

- Covariance matrix -

x	y

samples = N

mean of $X = \mu_x$

mean of $y = \mu_y$

step 1 : find μ_x and μ_y

step 2 : var of X → $\left[(\mu_x - x_1)^2 + (\mu_x - x_2)^2 + \dots + (\mu_x - x_n)^2\right] / N$

step 3 : var of y → ——————

$$cov(x,y) = \left[(\mu_x - x_1)\cdot(\mu_y - y_1) + (\mu_x - x_2)\cdot(\mu_y - y_2) + \dots (\mu_x - x_n)\cdot(\mu_y - y_n)\right] / (N-1)$$

$$\Sigma = \begin{bmatrix} var(x) & cov(x,y) & cov(x,z) \\ cov(x,y) & var(y) & cov(y,z) \\ cov(x,z) & cov(y,z) & var(z) \end{bmatrix}$$

PCA ⟶ covariance matrix (Σ)

→ eigenvalue and eigen vectors of Σ

$A.x = \lambda.x$ → normalize eigen vector → $e_1 = \begin{bmatrix} a \\ b \end{bmatrix}$ → $e_1 = \begin{bmatrix} \frac{a}{\sqrt{a^2+b^2}} \\ \frac{b}{\sqrt{a^2+b^2}} \end{bmatrix}$

→ derive dataset

x	y	P_1	P_2

$P_{11} = e_1^T \begin{bmatrix} x_1 - \mu_x \\ y_1 - \mu_y \end{bmatrix}$

$P_{12} =$ ——— ———

SVD Let $A = \begin{bmatrix} 3 & 3 & 2 \\ 2 & 3 & -2 \end{bmatrix}$

① $A.A^T = \begin{bmatrix} 3 & 3 & 2 \\ 2 & 3 & -2 \end{bmatrix} \cdot \begin{bmatrix} 3 & 2 \\ 3 & 3 \\ 2 & -2 \end{bmatrix} = \begin{bmatrix} 17 & 8 \\ 8 & 17 \end{bmatrix}$

$A.A^T - \lambda I = 0$

$w - \lambda I = 0$

↳ $\lambda^2 - 34\lambda + 225 = 0$

$(\lambda - 25)(\lambda - 9) = 0$

$\lambda_1 = 25$ $\lambda_2 = 9$

② $A.A^T - \lambda_1 I = \begin{bmatrix} 1 & -1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$ ($\lambda_1 \downarrow 25$)

$y = w' \cdot x$

↓

$w = [e_1\ e_2]$

regression → gradient descent

g(x) →

bayes →

bias - variance

normal dist $\frac{1}{2}$

---

scan 0,1

S = 1

terorist : 0,1

citizen : 0,1

$P(S=1 \mid T=1) = 95/100$

$P(S=0 \mid T=0) = 95/100$

$P(T=1 \mid S=1) = \dfrac{P(S=1 \mid T=1) \cdot P(T=1)}{P(S=1)}$

$P(T=1 \mid S=1) = \dfrac{95/100 \cdot 1/100}{1}$

$P(T=1) = P(T=1 \mid S=0) + P(T=1 \mid S=1)$

$99/100$

$P(S=1) = P(S=1 \mid T=1) \cdot P(T=1) + P(S=1 \mid T=0) \cdot P(T=0)$

$P(A=0 \mid B=0) = \%x \rightarrow P(A=1 \mid B=0) = \%(100-x)$

$P(A=1) = P(A=1 \mid B=0) \cdot P(B=0) + P(A=1 \mid B=1) \cdot P(B=1)$

Gradient Des

$$w^{k+1} = w^k - \alpha . \nabla g(w^k)$$

Linear $D_m = \frac{-2}{n} \sum_{i=0}^{n} x_i (y_i - \hat{y}_i)$

$$D_c = \frac{-2}{n} \sum_{i=0}^{n} (y_i - \hat{y}_i)$$

bias - variance

error = $bias^2$ + variance

$$E\left[(d-\theta)^2\right] = \underbrace{(E[d]-\theta)^2}_{bias} + E\left[(d-E[d])^2\right]$$

Gaussian

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right]$$

$$g_i(x) = \log\left[\frac{1}{\sqrt{2\pi\sigma_i^2}} \exp\left[-\frac{(x-\mu_i)^2}{2\sigma^2}\right] \cdot P(c_i)\right]$$

$$\frac{d}{dx}\left(a^x\right) = a^x . \ln a$$

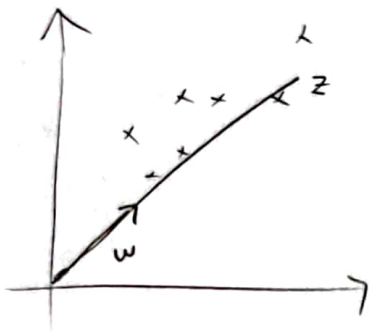
$$\frac{d}{dx}\left(\log_a g(x)\right) = \frac{g'(x)}{g(x).\ln a}$$

* $\frac{d}{dx}\left(e^{g(x)}\right) = e^{g(x)} . g'(x)$

* $\frac{d}{dx}\left(a^{g(x)}\right) = \ln(a) . a^{g(x)} . g'(x)$

After Midterm

PCA



$z^i = w^T \cdot x^i$, $\|w\| = 1$

arg max variance

$\text{argmax} \; \underbrace{E[(z - E(z))^2]}$

$E\left[(w^T x - \underbrace{E[w^T x]}_{w^T E[x]})^2\right]$ → x is variable, $w^T E[x] = w^T \mu$

$L(w,d) = E\left[(w^T x - E[w^T x])^2\right] - \lambda . w^T . w$

$= E\left[(w^T x - w^T \mu)^2\right] - \lambda . w^T . w$

$[w_1, w_2 \dots w_n] \cdot \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} x = w_1 x_1 + w_2 x_2 - \dots + w_n x_n = [x_1, x_2 \dots x_n] \cdot \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix} w$

w^T → x^T

Hatırlatma → $w^T . x = x^T . w$

$E\left[(w^T x - w^T \mu) . (w^T x - w^T \mu)\right]$

$E\left[w^T (x - \mu) \cdot (x^T w - \mu^T w)\right]$

$E\left[w^T (x - \mu)\,(x^T - \mu^T) . w\right]$

$w^T \underbrace{E\left[(x - \mu) . (x^T - \mu^T)\right]}_{\Sigma} w$

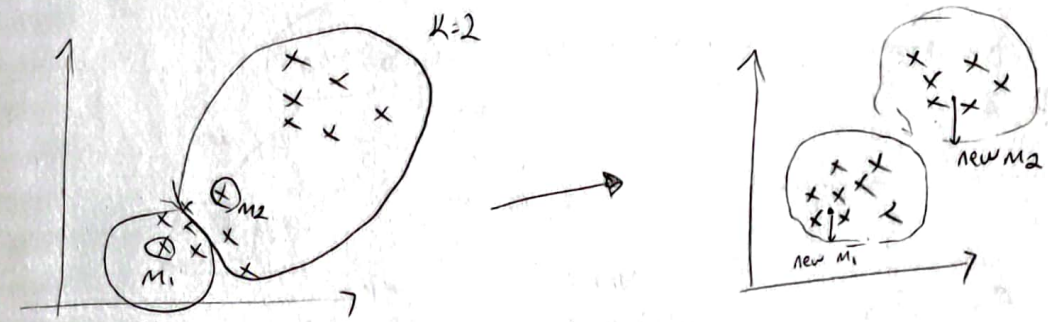
$$L(w,d) = w^T . \Sigma . w - \lambda . w^T . w$$

Unsupervised Learning

- unlabelled data
- discovers groups/clusters or patterns within the data
- dimensionality reduction, clustering → two most popular.

- classes vs clusters

K-means clustering



** It is an iterative algorithm not a closed form formula.

Initialization of m_i's:

- Random labelling (not recommended)
- mean + small random vectors
- Random data points as centers
- PCA and take the means of these groups

* multiple restart for more accurate results.

drawbacks of k-means

it may stuck in local minima

$$\frac{d}{dx}\left(a^{g(x)}\right) = a^{g(x)} \cdot \ln(a) \cdot g'(x)$$

$$\frac{d}{dx}\left(\log_a g(x)\right) = \frac{g'(x)}{g(x) \cdot \ln(a)}$$

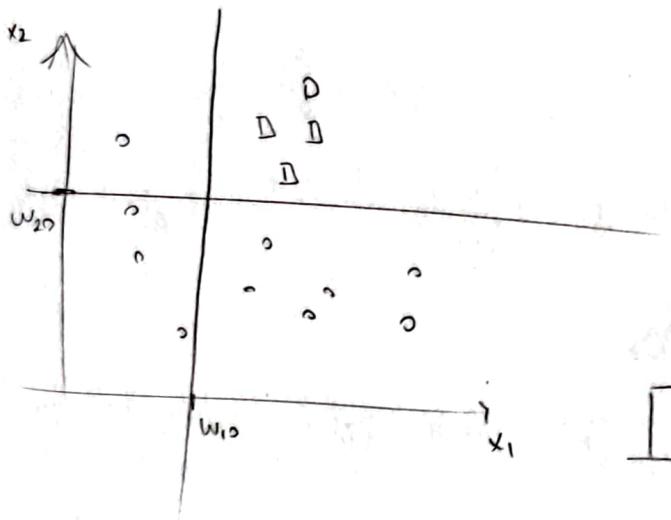
HATIRLATMA

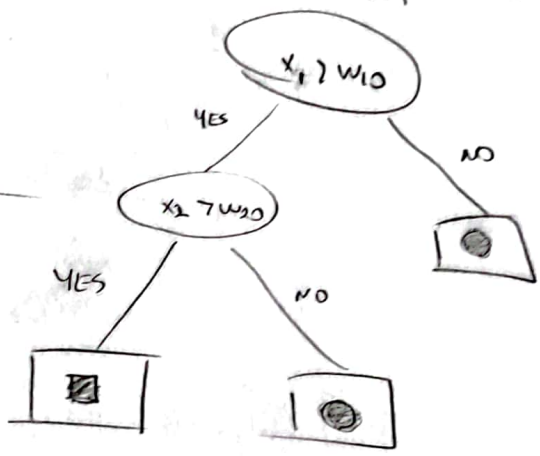
Anil K. Jain
youtube talk

Trees / Decision Trees

(classification trees)

↳ supervised





Splitting may vary → yes into
→ x=10, x=12, x=16
→ ——— N cases

Gini index = $P \cdot (-P)$ →

	x_1	x_2	x_3
+	3	∩	m
+	4	y	f
+	4	y	m
+	5	y	f
−	5	∩	f
−	4	∩	f
−	5	y	m
−	5	∩	m

x_1

3	4	5
1+	2+	1+
0−	1−	3−

Gini = $\frac{1}{1}$ $\frac{2}{3} \cdot \frac{1}{3}$ $\frac{1}{4} \cdot \frac{3}{4}$

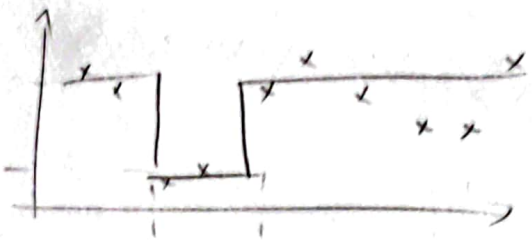
Impurity (expected value) $\frac{1}{1} \cdot \frac{1}{8} + \frac{2}{8} \cdot \frac{3}{8} + \frac{3}{16} \cdot \frac{4}{8}$

x_2

∩	y
1+	3+
3−	1−
$\frac{1}{4} \cdot \frac{3}{4}$	$\frac{3}{4} \cdot \frac{1}{4}$

Imp = $\frac{3}{16} \cdot \frac{4}{8} + \frac{3}{16} \cdot \frac{4}{8}$

Regression Trees



we choose a cluster pair that minimizes the Error f.

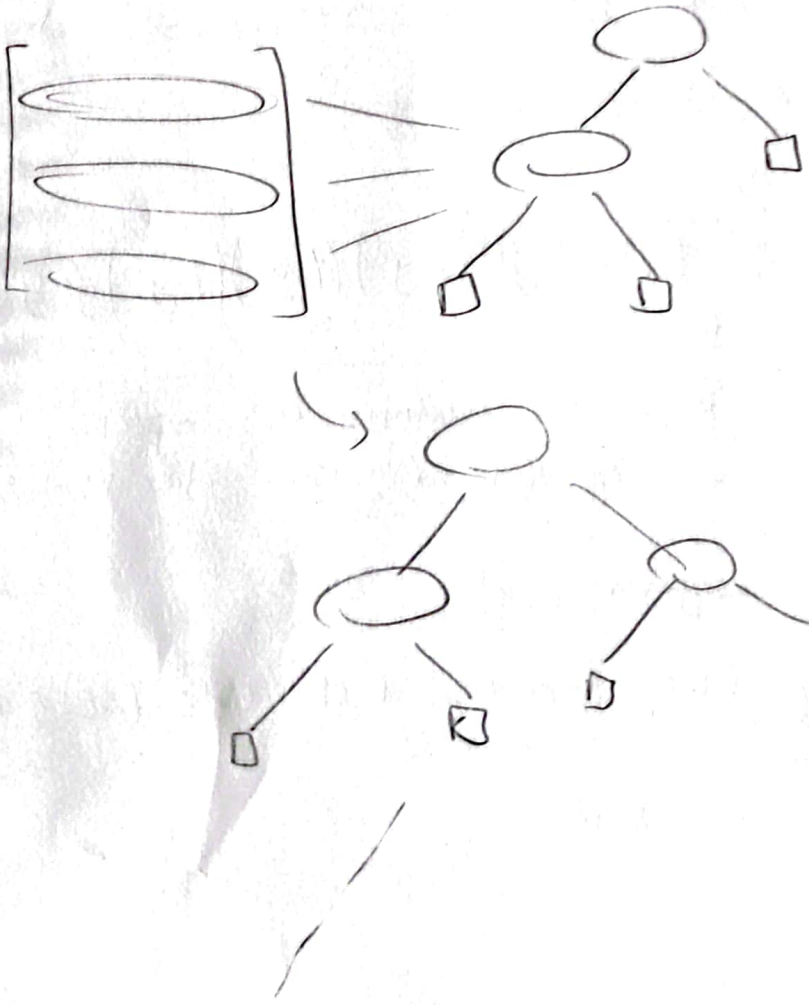
Pruning trees

after constructing tree, remove subtrees for better generalization. (avoid overfitting)

decrease variance → increase bias

Random Forest

Randomizing features and creating a model with them, then combining all of them to get a good estimation.

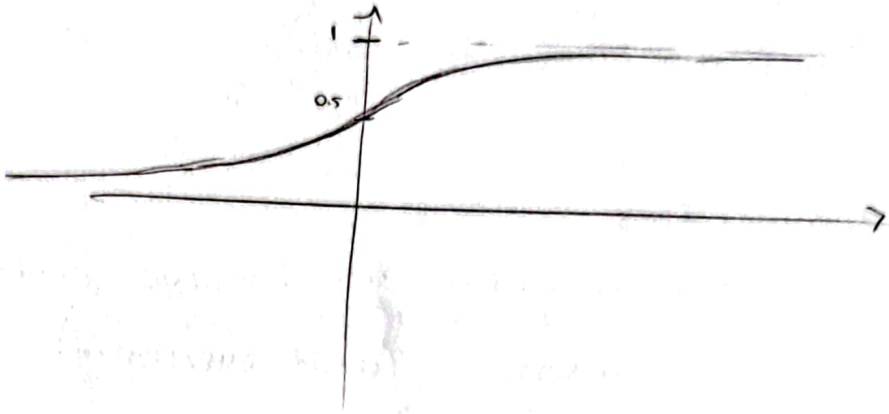


Let's say we have a new data we put our new data to all the decision trees and get the weighted result.

NEW WEEK !

(Logistic) Sigmoid function

$$g(s) = \frac{1}{1+e^{-s}}$$



$$g'(s) = \frac{e^{-s}}{(1+e^{-s})^2} = \text{---} = g(s).(1-g(s))$$

$$\prod (y^t)^{r^t} (1-y^t)^{(1-r^t)}$$

Bernoulli reminder

$$P(X=x) = \begin{cases} p & \text{for } x=1 \\ 1-p & \text{for } x=0 \end{cases}$$

x_1	x_2	r	y
		0	0
		0	0
		0	0
		0	0
		1	1
		1	1
		1	1

$$l(\theta) = (1-y^1)(1-y^2)(1-y^3)(1-y^4)\,y^5 \cdot y^6 \cdot y^7 \cdot y^8$$

If we can maximize this expression we can get a close classification.

$$\arg\max l(\theta)$$

To do this we use gradient descent. But this time for maximize.

$$w = w + \nabla w$$

$$E = \log \prod_{t=1}^{N} (y^t)^{r^t} \cdot (1-y^t)^{1-r^t}$$

$$y^t = \frac{1}{1+e^{-(w_0+w_1x_1+w_2x_2)}}$$

$$E = \sum_{t=1}^{N} r^t \log(y^t) + (1-r^t)\log(1-y^t) \rightarrow \text{maximize this (log likelihood)}$$

↳ or minimize -log likelihood

gradient descent

w_0, w_1, w_2 ← initialize small numbers around 0

$$w_0 = w_0 - \eta \frac{\partial E}{\partial w_0}$$

$$w_1 = w_1 - \eta \frac{\partial E}{\partial w_1}$$

$$\frac{\partial E}{\partial w_1} =$$

HATIRLATMA

$$\frac{d}{dx}(\log_a g(x)) = \frac{g'(x)}{g(x).\ln a}$$

$$\frac{d}{dx}(a^{g(x)}) = a^{g(x)}.g'(x).\ln a$$

$$-\sum \frac{r^t.(y^t)'}{y^t} + (1-r^t).\frac{(1-y^t)'}{1-y^t} = \frac{r^t y^t(1-y^t)}{y^t} + \frac{(1-r^t).(-y^t(1-y^t))}{1-y^t}$$

$$\sum (r^t - y^t)x_1^t$$

example

x_1	x_2	r
2	3	0
1	0	0
5	4	1
2	4	1

$w_0 = 0.1 \quad w_1 = 0,1 \quad w_2 = 0.05$

random initilize around 0

$\eta = 0,1$

$$y^1 = \frac{1}{1 + e^{-(0,1 + 0,1 \cdot x_1 + 0,05 \cdot x_2)}} = \frac{1}{1 + e^{-0,45}} = 0,61$$

$$y^2 = \frac{1}{1 + e^{-0,2}} = 0,54 \qquad y^3 = \frac{1}{1 + e^{-98}} = 0,68$$

$$y^4 = \frac{1}{1 + e^{-0,5}} = 0,62$$

$$w = w - \eta \nabla w$$

$$w = w - \eta \sum (r^t - y^t) \cdot x_1^t$$

$$w_1 = w_1 - \eta \left[(r^1 - y^1) x_1^1 + (r^2 - y^2) \cdot x_1^2 + (r^3 - y^3) \cdot x_1^3 + (r^4 - y^4) x_1^4 \right]$$

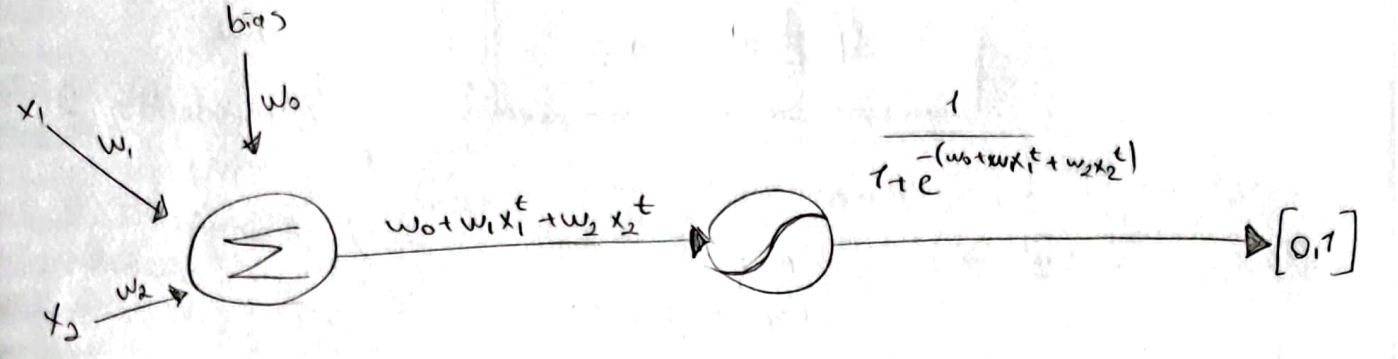
$$w_1 = w_1 - 0,1 \cdot \left[(0 - 0,61) 2 + (0 - 0,54) \cdot 1 + (1 - 0,68) \cdot 5 + (1 - 0,62) \cdot 2 \right.$$

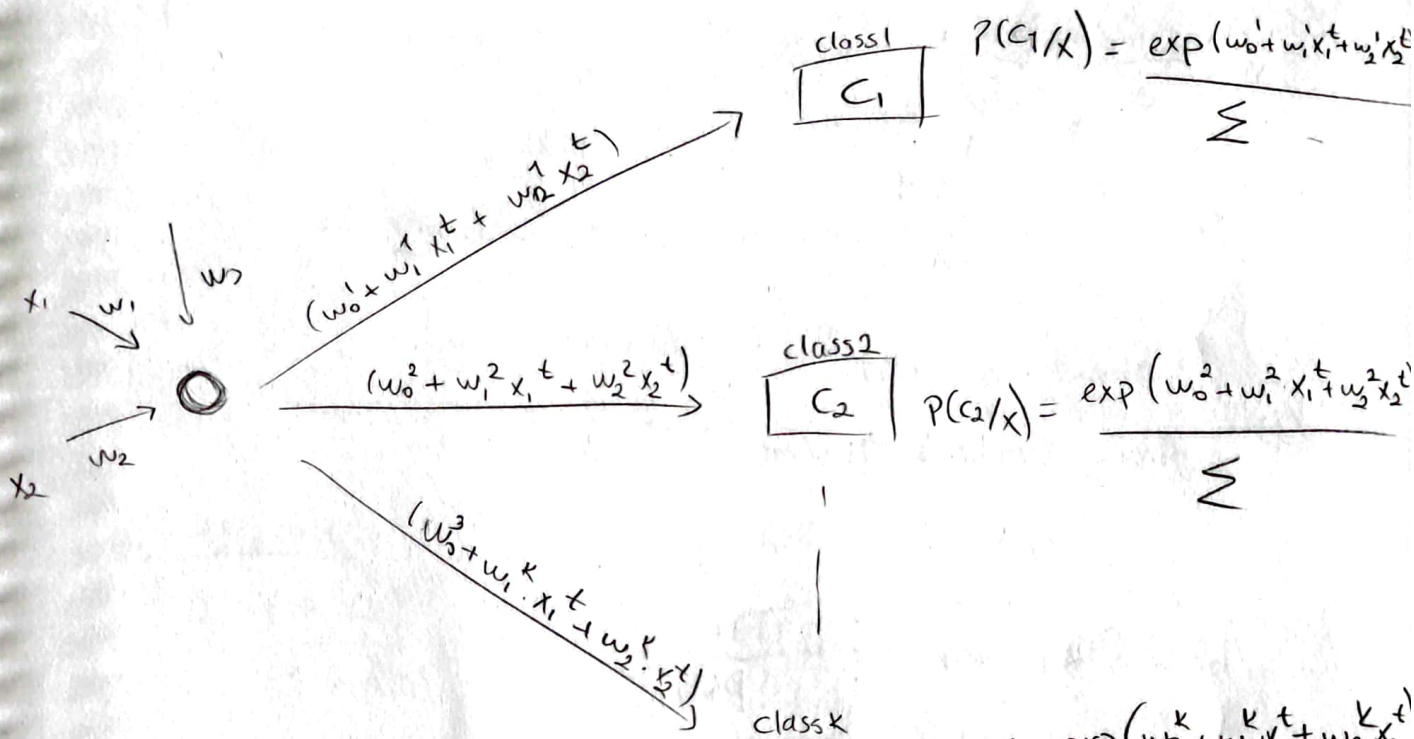
new-w_1 $0,04 = 0,1 - \frac{0,1 \cdot 0,6}{0,06}$

and same goes for w_2

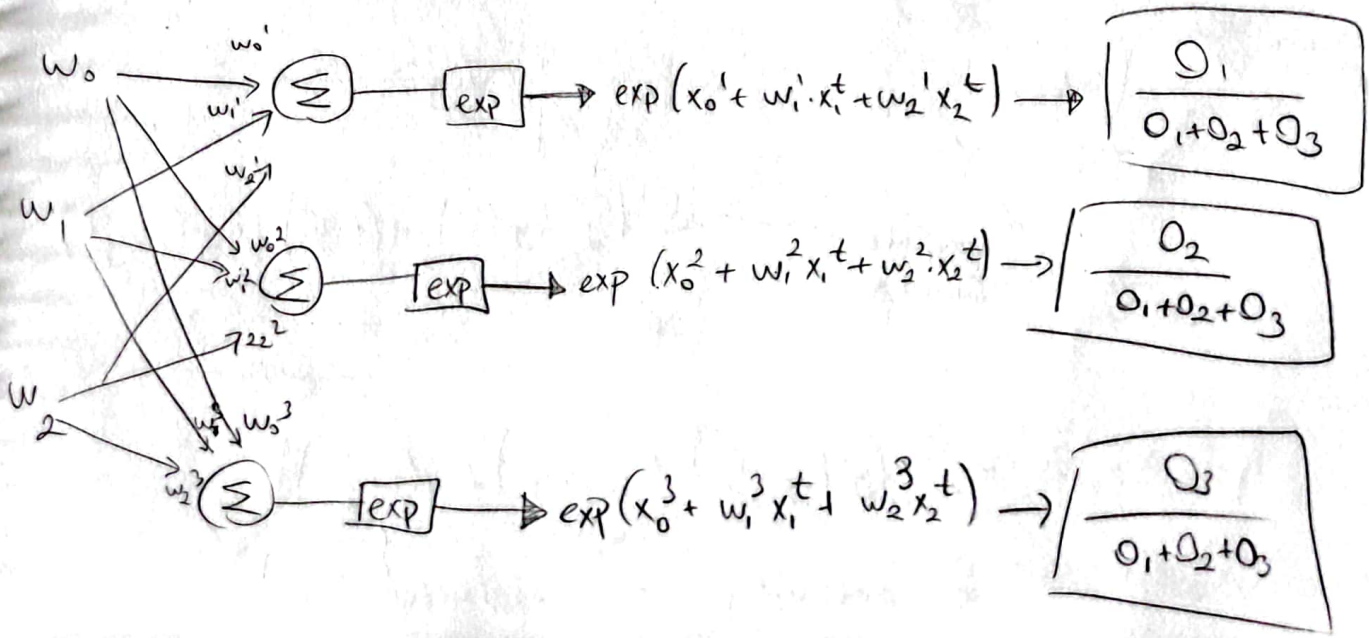
w_0 goes same as well but there is no 'x' term.

goes with converge





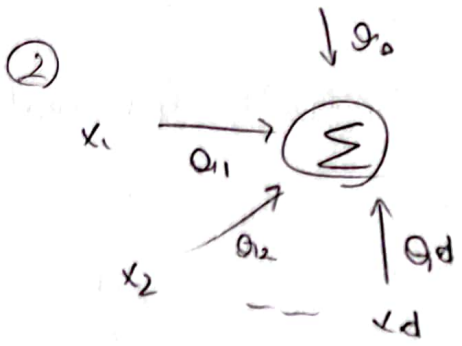
another representation



WEEK 13

Introduction 2 ML "Dimitri —" youtube

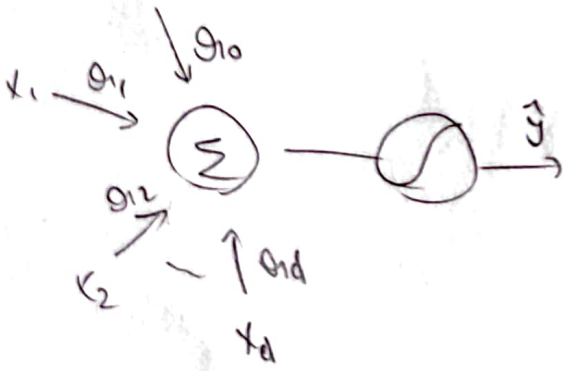
① $\sum_{i=1}^{N} (\hat{y}^i - y^i)^2$ = MSE (error function)

② 

③ Apply Gradient Descent

$\theta_{10}, \theta_{11}, \theta_{12}, \ldots, \theta_{1d}$

$\theta_{1k+1} = \theta_{1k} - h \cdot \frac{\partial E}{\partial \theta_{1k}}$



$\prod_{i=1}^{N} (\hat{y}^i)^{y^i} \cdot (1-\hat{y}^i)^{1-y^i}$

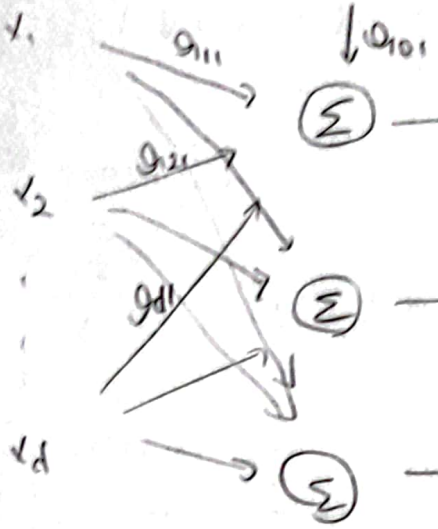
$\downarrow \log$

maximize likelihood $\leftarrow \sum \left(y^i \cdot \log(\hat{y}^i) + (1-y^i) \log(1-\hat{y}^i) \right)$

OR

$-\sum \left(y^i \log(\hat{y}^i) + (1-y^i) \cdot \log(1-\hat{y}^i) \right)$

cross entropy (minimize this)



$$O_1 = \theta_{101} + \theta_{111} \cdot x_1 + \theta_{201} \cdot x_2 + \dots + \theta_{d1} \cdot x_d$$

$$O_2 = \theta_{102} + \theta_{112} \cdot x_1 + \theta_{122} \cdot x_2 + \dots + \theta_{d2} \cdot x_d$$

$$O_3 = \theta_{103} + \theta_{113} \cdot x_1 + \theta_{123} \cdot x_2 + \dots + \theta_{d3} \cdot x_d$$

to normalize these output values:

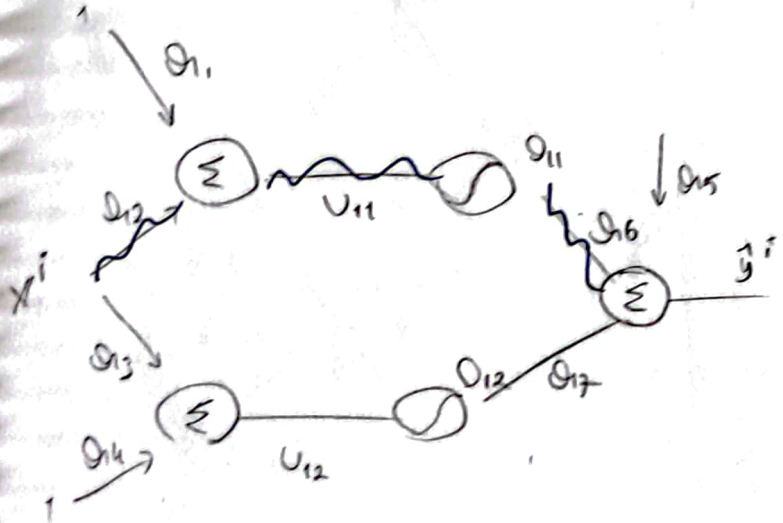
$$\hat{y}_k = \frac{\exp(O_k)}{\sum_{i=1}^{N} \exp(O_i)} \qquad \text{(softmax function)}$$

In this case

$$\hat{y}_1 = \frac{\exp(O_1)}{\sum_{i=1}^{3} \exp(O_i)}$$

$$\prod_{t=1}^{N} \prod_{i=1}^{3} (\hat{y}_i^t)^{y_i^t} = \text{likelihood}$$

$$mse = \sum (\hat{y}^i - y^i)^2$$



$$\hat{y} = \theta_5 + \theta_6 \cdot O_{11} + \theta_7 \cdot O_{12}$$

$$O_{11} = \frac{1}{1+e^{-u_{11}}}$$

$$O_{12} = \frac{1}{1+e^{-u_{12}}}$$

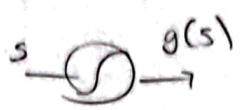
$$u_{11} = \theta_1 + \theta_2 \cdot x^i$$

$$u_{12} = \theta_4 + \theta_3 x^i$$

$$\frac{\partial E}{\partial \theta_{12}} = \frac{\partial E}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial O_{11}} \cdot \frac{\partial O_{11}}{\partial u_{11}} \cdot \frac{\partial u_{11}}{\partial \theta_{12}}$$ (chain rule)

$\sum(\hat{y}^i - y^i)$ $\quad O_{16}$ $\quad O_{11}\cdot(1-O_{11})$ $\quad x^i$

REMINDER

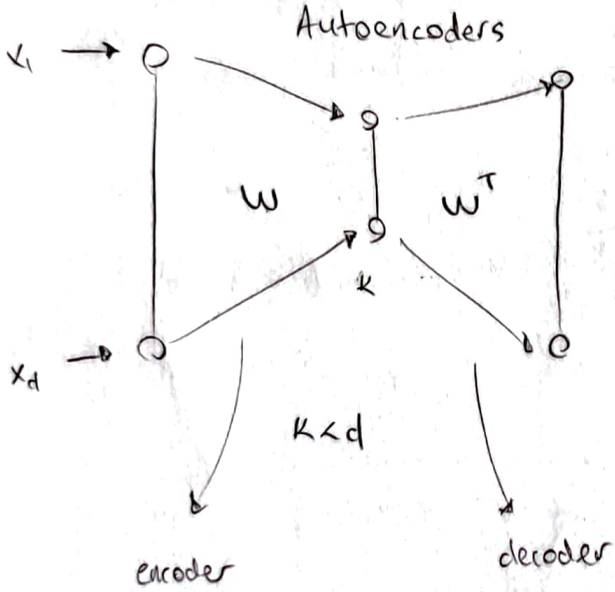


$$\frac{\partial g(s)}{\partial s} = g'(s)$$

$$g(s) = \frac{1}{1+e^{-s}} \qquad g'(s) = \frac{e^{-s}}{(1+e^{-s})^2}$$

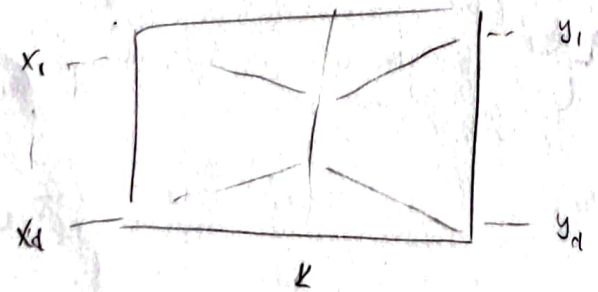
$$g'(s) = g(s)\cdot(1-g(s))$$

1 output → linear regression ($\hat{y}$)



$[x]_{100,10}$

N d

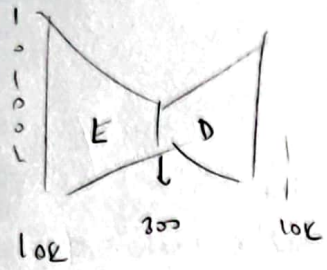


K<d

	y_1 y_2 y_3
x_1 x_2 x_3	x_1 x_2 x_3

- dimensionality reduction

Word 2 Vec ⟶ [ ]$_{300\ vector}$

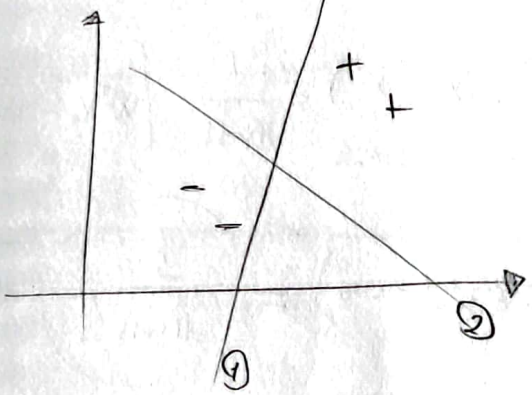


I have (lost) the will to live...

translation → RNN

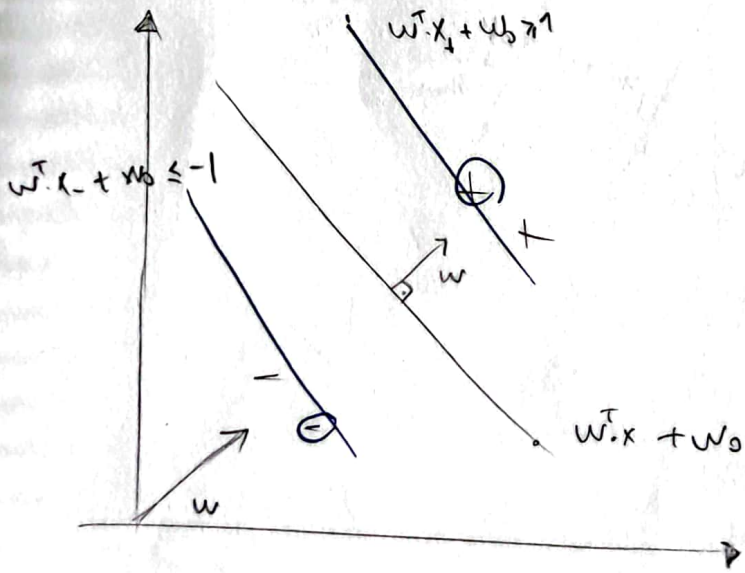
one hot encoding attention

Support Vector Machines -SVM-



2 is better

we try to maximize the distance between boundry and datas.



$$w^T.x + w_0 \quad (b)$$

$$[---].\begin{bmatrix} | \\ x \\ | \end{bmatrix} + b$$

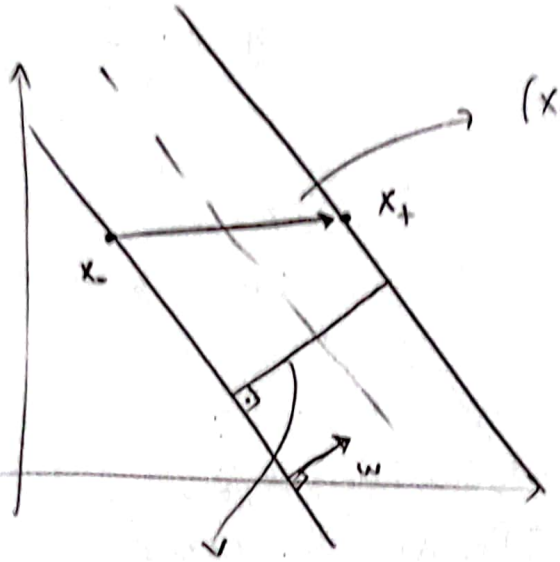
w vector will be perpendicular to decisi boundry.

$w^T.x_+ + w_0 > 0$

$w^T.x_- + w_0 < 0$

by definition our dataset is seperable.

$y^i (w^T \cdot x^i + w_0) \geq 1$

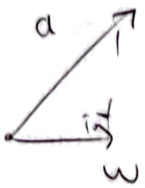


$w^T \cdot x_+ + w_0 = 1$

$w^T \cdot x_- + w_0 = -1$

project $(x_+ - x_-)$ vector onto w vector

maximize DISTANCE

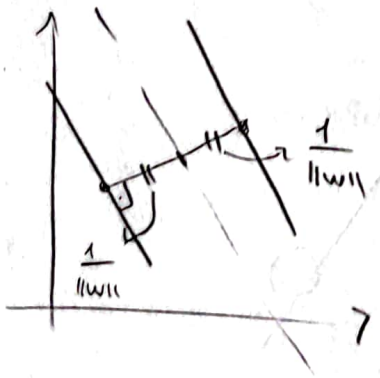


$\Rightarrow \frac{a^T \cdot w}{\|w\|}$

$$\text{Distance} = \frac{w^T}{\|w\|} \cdot (x_+ - x_-) = \frac{1}{\|w\|} \cdot \left[w^T x_+ - w^T x_- \right]$$

$$= \frac{1}{\|w\|} \cdot (1 - w_0 + 1 + w_0) = \frac{2}{\|w\|}$$

so maximize $\frac{1}{\|w\|} = \frac{1}{\sqrt{w^T \cdot w}}$



or

minimize $w^T \cdot w$

$$\frac{1}{2} w^T \cdot w = \frac{1}{2} \|w\|^2$$

to get rid of 2 when taken derivative

Lagrange

Lagrange

$$\mathcal{L}_P = \frac{1}{2}\|w\|^2 - \sum \alpha_i \underbrace{\left[y^i (w^T x^i + w_0) - 1\right]}_{y^i(w^T x^i + w_0) \geq 1}$$

$\alpha_i > 0$

$$= \frac{1}{2}\|w\|^2 - \sum (\alpha_i y^i w^T x^i + \alpha_i y^i w_0 - \alpha_i)$$

minimize $\frac{1}{2} w^T w$ (w)

subject to $y^i(w^T x^i + w_0) \geq 1$

$[w_0 \; \dots \; w_d] \begin{bmatrix} x_0 \\ \vdots \\ x_d \end{bmatrix}$ → NO w

$w_0 x_0 + w_1 x_1 + \dots + w_d x_d$

derivative with respect to w ⇒ x^i

$$\frac{\partial \mathcal{L}}{\partial w} = 0 = w - \sum \alpha_i y^i x^i$$

i) $w = \sum \alpha_i y^i x^i$

ii) $\frac{\partial \mathcal{L}}{\partial w_0} = \sum \alpha_i y^i = 0$

LAST WEEK :)

Instead of minimizing $\mathcal{L}_P$, people maximize $\mathcal{L}_D$

$$\mathcal{L}_D = \frac{1}{2}\underbrace{\left(\sum_i \alpha_i y^i x^{(i)}\right)^T}_{a^T} \cdot \underbrace{\left(\sum_j \alpha_j y^j x^{(j)}\right)}_{a} - \underbrace{\sum \left[\alpha_i y^i x^{(i)}\right]^T}_{a^T} \cdot \underbrace{\sum \left[\alpha_j y^j x^{(j)}\right]}_{a} + \sum_i \alpha_i y^i w_0 - \sum \alpha_i$$

$$= \boxed{\frac{-1}{2} \sum_i \sum_j \alpha_i \alpha_j y^{(i)} y^{(j)} (x^{(i)})^T \cdot x^{(j)} + \sum \alpha_i}$$

exp)

$y=1$, $\begin{bmatrix} 1 \\ 1 \end{bmatrix}$ - $\begin{bmatrix} -1 \\ 1 \end{bmatrix}$

$y=-1$, $\begin{bmatrix} -1 \\ -1 \end{bmatrix}$

$$\alpha_1 \cdot 1 \cdot \begin{bmatrix} 1 \\ 1 \end{bmatrix} + \alpha_2 \cdot 1 \begin{bmatrix} -1 \\ +1 \end{bmatrix} + \alpha_3 (-1) \cdot \begin{bmatrix} -1 \\ -1 \end{bmatrix} = 0$$

$\sum_i \alpha_i y^i x^{(i)} = 0$

$$\left(\alpha_1 \begin{bmatrix}1\\1\end{bmatrix}^T + \alpha_2 \begin{bmatrix}-1\\1\end{bmatrix}^T - \alpha_3 \begin{bmatrix}-1\\-1\end{bmatrix}^T\right)\left(\alpha_1 \begin{bmatrix}1\\1\end{bmatrix} + \alpha_2 \begin{bmatrix}-1\\1\end{bmatrix} - \alpha_3 \begin{bmatrix}-1\\-1\end{bmatrix}\right)$$

$$= \alpha_1^2 [1\ 1]\begin{bmatrix}1\\1\end{bmatrix} + \alpha_1\alpha_2 [1\ 1]\begin{bmatrix}-1\\1\end{bmatrix} - \alpha_1\alpha_3 [1\ 1]\begin{bmatrix}-1\\-1\end{bmatrix} -$$
$$+ \alpha_2\alpha_1 [-1\ 1]\begin{bmatrix}1\\1\end{bmatrix} + \alpha_2^2 [-1\ 1]\begin{bmatrix}-1\\1\end{bmatrix} - \alpha_2\alpha_3 [1\ 1]\begin{bmatrix}-1\\-1\end{bmatrix}$$
$$- \alpha_3\alpha_1 [-1\ -1]\begin{bmatrix}1\\1\end{bmatrix} - \alpha_2\alpha_3 [-1\ -1]\begin{bmatrix}-1\\1\end{bmatrix} + \alpha_3^2 [-1\ -1]\begin{bmatrix}-1\\-1\end{bmatrix}$$

$$= 2\alpha_1^2 + 0.\alpha_1\alpha_2 - 2\alpha_1\alpha_3 + 0.\alpha_1\alpha_2 + 2\alpha_2^2 + 0\alpha_1\alpha_3 + 2\alpha_1\alpha_3 + 0\alpha_2\alpha_3 + 2\alpha_3^2$$
$$+ \sum_i \alpha_i$$

$$L_D = 2\alpha_1^2 + 2\alpha_2^2 + 2\alpha_3^2 + 4\alpha_1\alpha_3 + \alpha_1\alpha_2\alpha_3 \longrightarrow \alpha_1 + \alpha_2 - \alpha_3 = 0$$
$$\alpha_3 = \alpha_1 + \alpha_2$$

to find maximum

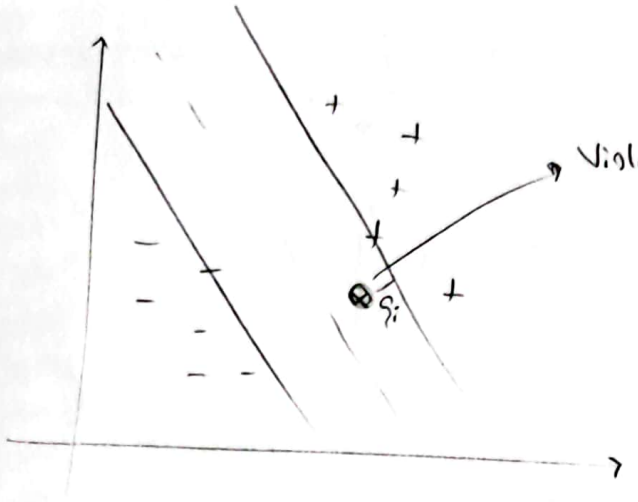
coordinate ascent

** For most of the points α's will be zero,
non-zero α values ⟶ support vectors

---

another example on the book ML Making sense of Data
"Peter Flach"

example 7.5, p. 215



$y^i(w^T x^i + w_0) \geq 1 - \xi_i$

$\sum \xi_i$

sum of violations

minimize $\frac{1}{2} w^T . w + C . \sum \xi_i$

a constant to make fine tuning on violations

Now we have this:

$w = \sum \alpha_i y^i x^{(i)}$

$\sum \alpha_i y^i = 0$

$\boxed{0 \leq \alpha_i \leq C}$ → new constraint

$\alpha_i^2 \underbrace{[-1 \;\; 2]}_{2\text{-dim}} \begin{bmatrix} -1 \\ 2 \end{bmatrix} + \cdots$

$x = \begin{bmatrix} -1 \\ 2 \end{bmatrix} \begin{matrix} \rightarrow x_1 \\ \rightarrow x_2 \end{matrix}$

$-\frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y^i y^j \underbrace{(x^i)^T . x^j}_{(z^i)^T (z^j)} + \sum_i \alpha_i$

$(\phi(x^i))^T . \phi(x^j)$ → Let's rename → $\phi^T(x) . \phi(y)$

$\phi(x) = \begin{bmatrix} 1 \\ \sqrt{2} x_1 \\ \sqrt{2} x_2 \\ \sqrt{2} x_1 x_2 \\ x_1^2 \\ x_2^2 \end{bmatrix} = z$

$\phi^T(x) . \phi(y) = 1 + 2x_1 y_1 + 2x_2 y_2 + 2x_1 x_2 y_1 y_2 + x_1^2 y_1^2 + x_2^2 y_2^2$ → polynomial

$K(x, y) = (x^T y + 1)^2 = \phi^T(x) . \phi(y)$

Aim: Data may not be seperable in lower dim-space. Therefore, we try higher dim-spaces but efficient

Here $\phi(x)$ is a function that takes a 2-dim matrix and maps it into a 6-dim one. Dim++

$$-\frac{1}{2}\sum_i\sum_j \alpha_i \alpha_j y^i y^j K(x^i, x^j) + \sum \alpha_i$$

maximize

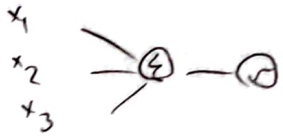
Kernel

mercer conditions

Online Session

Artificial neural networks take their inspiration from brai

1960 - Multilayer Perceptron



1986 - Back Propagation

1989 - Convolutional Neural Network

1990 - 1994 - Neural Nets in the Wild (Support vector Machines ↑)

1995 - long short term memory

1998 - 2005 - more N.N in the wild

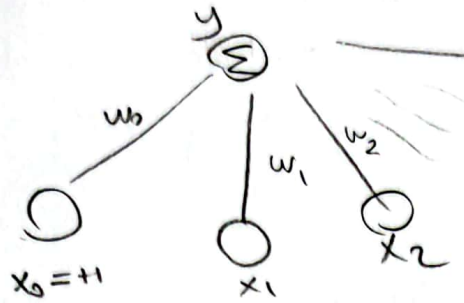
2005 - 2010 - Banishment (not enough data and CPU power)

2010 - Renaıe : Deep Learning

2013 - CNN + GPU + ImageNet

2015 - Alphago

Perceptron

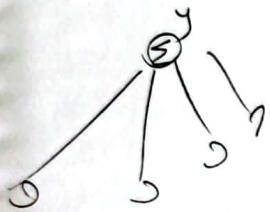


$$y = \sum_{j=1}^{d} w_j x_j + w_0 = w^T x$$

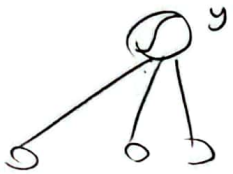
(1962)

what perceptron does

regression : $y = wx + w_0$

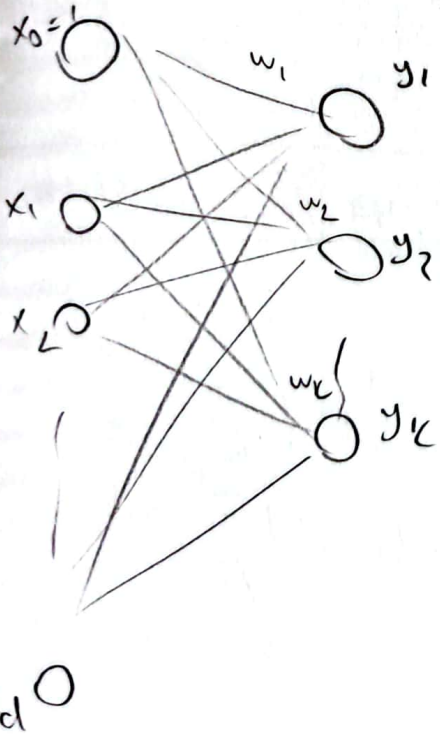


classification $y = 1(wx + w_0 > 0)$



$y = \text{sigmoid}(x) = \frac{1}{1+\exp[-w}$

K outputs



regression: $y_i = \sum_{j=1}^{d} w_{ij} \cdot x_j + w_{i0} = w_i^T$

classification

$$o_i = w_i^T x$$

$$y_i = \frac{\exp o_i}{\sum_k \exp o_k}$$

Training → to train our weights (w)

online → instances seen one by one

batch → instances seen as whole

gradient descent

$$w = w - \eta \frac{\partial E}{\partial w}$$

stochastic gradient descent → update after a single pattern

Regression

$$E = \sum_{t=1} (r^t - y^t)$$

r^t: actual, y^t: predicted

$$\rightarrow \Delta w_j^t = \eta (r^t - y^t) x_j^t$$

for every instance $E^t(w \mid x^t, r^t) = \frac{1}{2} \cdot (r^t - y^t)^2 = \frac{1}{2} [r^t - (w^t x^t)]^2$

Classification

E → cross entropy

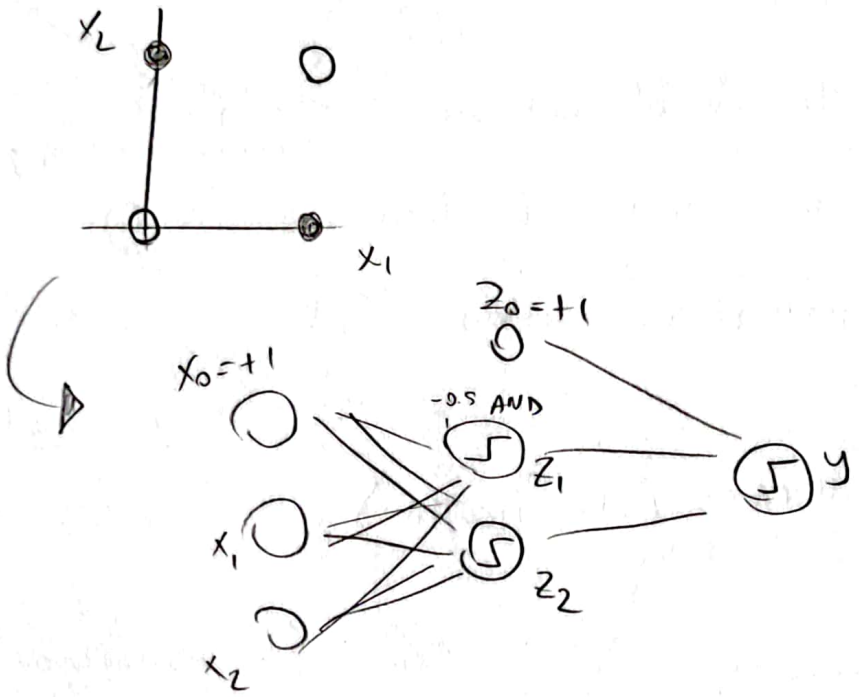
$$\log \prod (y^t)^{r^t} \cdot (1-y^t)^{1-r^t}$$

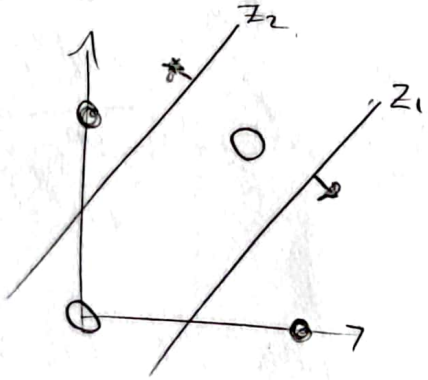
stochastic gradient descent

$$E^t(w \mid x^t, r^t) = -r^t \log y^t - (1-r^t) \log (1-y^t)$$

$$\Delta w_{ij}^t = \eta (r_i^t - y_i^t) x_j^t$$

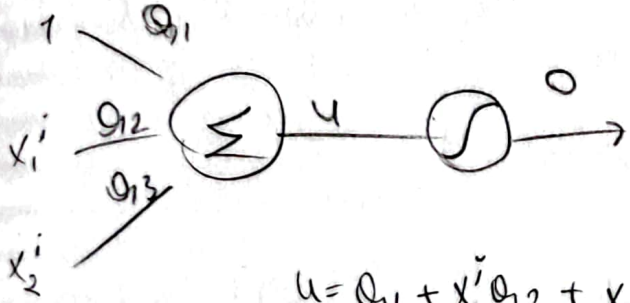
a linear model may not correctly classify 4 data points ⟶ XOR problem





more layers help to classify better.

MLP - 1 neuron



$$\hat{y}^i = P(y^i = 1 / x^i)$$

↳ probability of $y^i = 1$ given that x^i

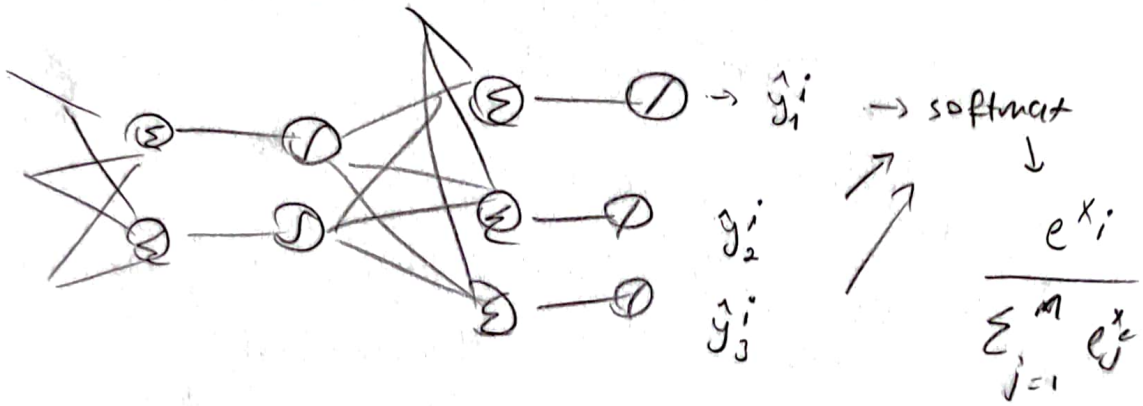
$$u = \theta_1 + x_1^i \theta_2 + x_2^i \theta_3$$

$$o = \frac{1}{1+e^{-u}} = \hat{y}^i$$

if MLP ends with → a sigmoid function → classification
↘ a Σ → regression

- Organize the neural network
- Define an error function → MSE / cross entropy
- Update parameters using G.D.

MLP – Multi class (classification)

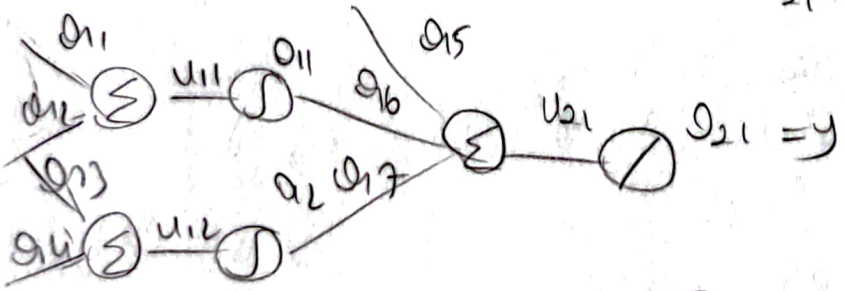


Backpropagation

$E = \frac{1}{2}(y-r)^2$ $y = \theta_{21} = u_{21}$

$u_{21} = \theta_5 + \theta_6 \cdot \theta_{11} + \theta_7 \theta_{12}$

regression

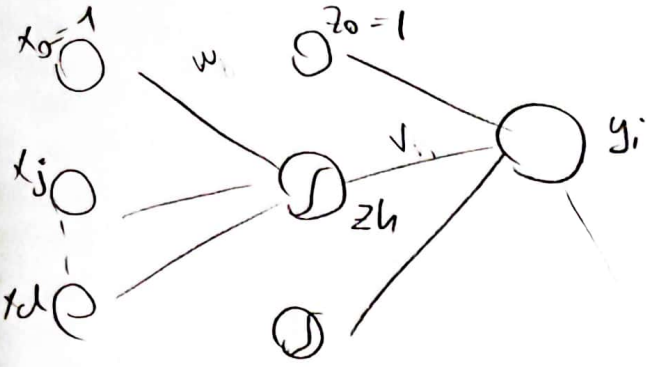


$$\Delta\theta_6 = \frac{\partial E}{\partial \theta_6} = \frac{\partial E}{\partial y} \cdot \frac{\partial y}{\partial \theta_{21}} \cdot \frac{\partial \theta_{21}}{\partial u_{21}} \cdot \frac{\partial u_{21}}{\partial \theta_6}$$

USE CHAIN RULE

$(y-r)\,\theta_{11}$ $\quad y \quad 1 \quad 1 \quad \theta_{11}$

Multilayer Perceptrons



regression

$E(w,v) = \frac{1}{2}(r - \hat{y})^2$

unknowns

$\hat{y} = v \cdot z$ $\quad \frac{\partial E}{\partial v} = (r - \hat{y}) \cdot z$

$\frac{\partial E}{\partial v} = \frac{\partial E}{\partial y} \cdot \frac{\partial y}{\partial v}$ ↗

General formula

$\Delta v_h = \sum (r^t - y^t) z_h^t$

$\frac{\partial E}{\partial w} = \frac{\partial E}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial z} \cdot \frac{\partial z}{\partial w}$

$(r - \hat{y}) \quad v \quad z(1-z)x \quad w$

$z \rightarrow$ sigmoid

$z' = z(1-z)$

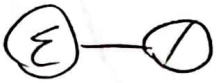
$\Delta w_{hj} = -\eta \frac{\partial E}{\partial w_{hj}}$

$\Delta w_{hj} = \eta \sum_t (r^t - y^t) v_h \cdot z_h^t \cdot (1 - z_h^t) \cdot x_j^t$

Multi output MLP

MLP with 2 hidden layers

Summary

① define problem — regression → 

classification → 

② MSE $\frac{1}{2}(r - \hat{y})^2$

$\prod \hat{y}^r (1-\hat{y})^{1-r} \rightarrow -[r \log y + (1-r) \log(1-y)]$

cross entropy