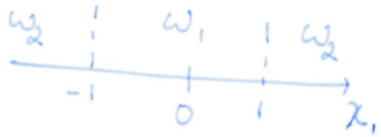
L12 : 24.06.2022

B Other mapping functions

E4.13 : Feature mapping

(a)

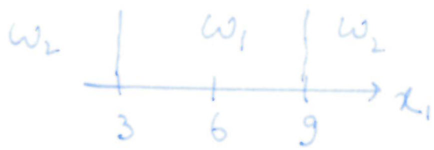


not linearly separable

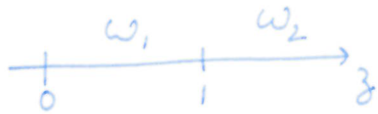
Let $z = x_1^2 \geq 0$

linearly separable

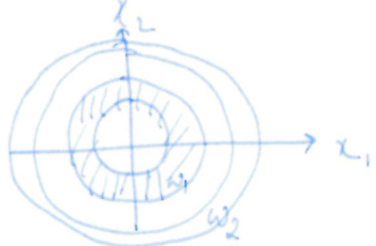
(b) dataset 3



$z = (x_1 - 6)^2$

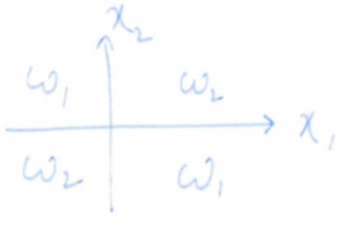


(c) dataset 2

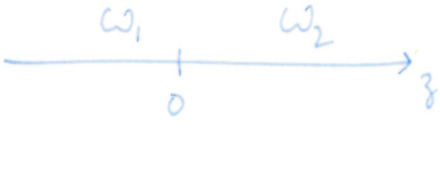


$z = \sqrt{x_1^2 + x_2^2}$

(d) XOR



$z = x_1 x_2$



Pro and cons of feature engineering / mapping:

+) same learning algorithm as for LDF in $\underline{z}$

+) more general decision regions in $\underline{x}$

→) difficult to find a good feature mapping for $d > 3$

- often polynomial used

−) huge dimension d'

A polynomial discriminant function can be formulated as a LDF:

$$f_j(\underline{x}) = \underline{u}_j^T \underline{z} \text{ with}$$

$$\underline{z} = [1, \quad x_1 \dots \quad x_d, \; x_1^2, \; x_1 x_2 \dots]^T$$

$$u_j = [w_{j,0} \quad w_{j,1} \dots \quad w_{j,d}, \; w_{j,11} \quad w_{j,12} \dots]^T$$

$x \in \mathbb{R}^d$: original feature space

$z \in \mathbb{R}^{d'}$: mapped feature space

eg $d = 100$

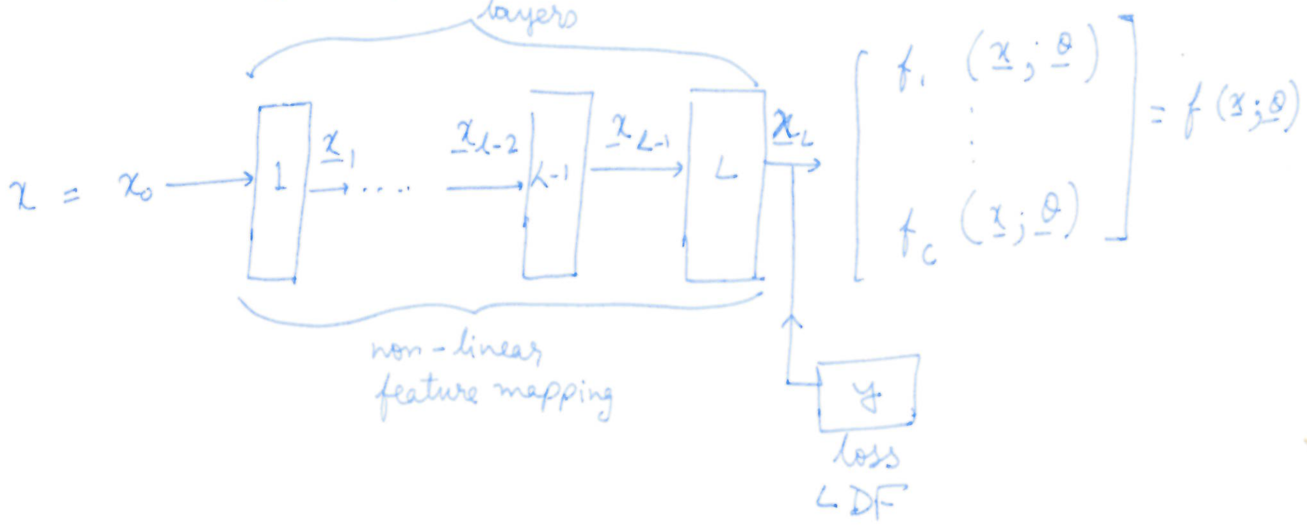
	LDF	$x_k x_l$	$x_k x_l x_m$
d'	$100+1=101$	$O(100^2)$	$O(100^3)$

→ • higher computational complexity

• not enough training samples → overfitting

4.5.3 Neural Network (NN)

Traditional NN



layers:

$$\underline{x}_l = \phi_l\big(\underbrace{\underline{\underline{W}}_l \underline{x}_{l-1} + \underline{b}_l}_{LDF}\big) \quad ; \quad 1 \le l \le L,$$

non-linear ~~de~~ activation f.?

$$\phi_L(\underline{a}) = \underline{a}$$

Training:

loss $l(f(\underline{x};\underline{\theta}), y)$ eg L2 norm $\hat{=}$ LS

cost f.: $\mathcal{L}(\underline{\theta}) = \frac{1}{N}\sum_{n=1}^{N} l\left(f(\underline{x}(n);\underline{\theta}); \underline{y}(n)\right)$

$$\min_{\underline{\theta}} \mathcal{L}(\underline{\theta})$$

NN: automatically learnt non linear feature mapping

+

learned LDF.

4.5.4 Support Vector Machine (SVM)

Vapnik, 1996

Basic idea:

- a binary classifier: $y_n \in \{-1, 1\}$ instead of ω_1, ω_2
- use a LDF $f(\underline{x}) = \underline{w}^T \underline{x} + w_0$
- estimated class $\hat{y}(x) = \text{sign}(f(\underline{x})) \in \{-1, 1\}$

Advanced ideas

- non-linear-feature mapping

$$\underline{z} = \phi(\underline{x}) \text{ instead of } \phi_L(\,) \text{ in NN}$$

  - kernel trick

- training
  - maximum margin instead of LS in NN
  - convex optimization problem instead of non-convex in NN

4.5.4.1 Maximum margin

4-61

Assume $S = \{ (\underline{x}_n, y_n), 1 \leq n \leq N \}$ linearly separable

i.e. $\gamma_n > 0$

$\Rightarrow$ infinite number of LDFs $f(\underline{x})$ with

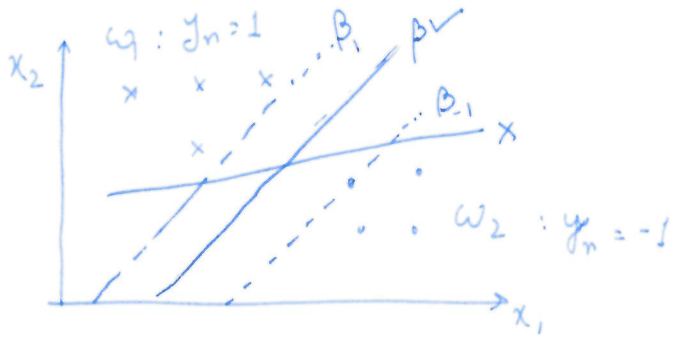
$\gamma (f(\underline{x}), S) > 0$

Which one ?

NN : LS - Least Squares

SVM : maximum margin

$$\max_{\underline{w}, w_0} \gamma (f(\underline{x}), S)$$



$\beta_1, \beta, \beta_{-1}$: 3 parallel hyperplanes

- $\beta = \{ \underline{x} \mid f(\underline{x}) = 0 \}$: desired decision boundary
- β_1, β_{-1} = 2 limit hyperplane with the same margin $\gamma > 0$ to β

β_1 : passes through the closest $\underline{x}_n$ from ω_1

β_{-1} : " " " " " ω_2

The pts on β_1, β_{-1} are called <u>support vectors</u>

$$SV = \{ n \mid \underline{x}_n \text{ on } \beta_1 \text{ or } \beta_{-1} \}$$

Scaling of $f(\underline{x})$:

If $f(\underline{x})$ is a LDF, then $\alpha f(\underline{x})$ with $\alpha > 0$ is also a LDF

Choose α such that $|f(\underline{x}_n)| = 1 \quad \forall n \in SV$

- $n \in SV$: margin $\gamma_n = y_n \cdot \frac{f(x_n)}{\|\underline{\omega}\|} = \frac{1}{\|\underline{\omega}\|}$
- $n \notin SV$: margin $\gamma_n > \frac{1}{\|\underline{\omega}\|}$
- $\gamma(f(\underline{x}), S) = \min\limits_n \gamma_n = \frac{1}{\|\underline{\omega}\|}$

<u>Maximum margin training:</u>

$$\max \gamma \text{ or } \min \|\underline{\omega}\| \text{ or } \min_{\underline{\omega}, \omega_0} \frac{1}{2} \|\underline{\omega}\|^2$$

$$\text{s.t. } y_n f(\underline{x}_n) = y_n (\underline{\omega}^T \underline{x}_n + \omega_0) \geq 1 \quad \forall n$$

in comparision to LS learning in NN:

\+ SVM less sensitive to statistical variations in $\underline{x}$

\+ " " " to noise in $\underline{x}$

$\Rightarrow$ lower generalization error rate

\+ perfect classification for linearly separable data

\+ Convex optimization ; no local minima, global convergence

+ only support vectors determine $f(\underline{x})$,

"forget" all non-support vectors

→ reduced complexity

4-62

4-63