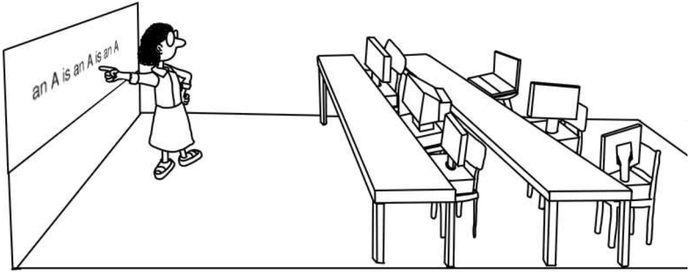
Machine Learning

Krystian Mikolajczyk & Deniz Gunduz

Department of Electrical and Electronic Engineering
Imperial College London



Machine Learning - Part 3 Summary

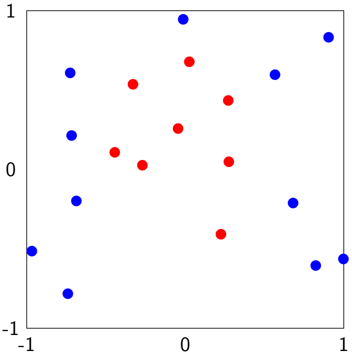
Department of Electrical and Electronic Engineering

Imperial College London

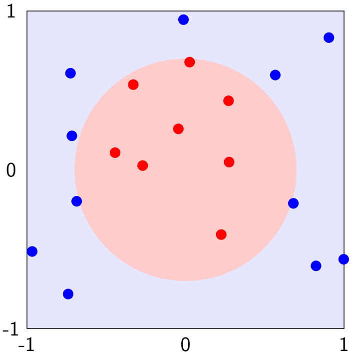
- Non-linear feature transform: polynomial, Legendre
- Overfitting/underfitting: match the hypothesis class to data
- Structural risk minimization
- Regularisation: L_2, L_1 ...
- Validation

Limits of Linear Representations

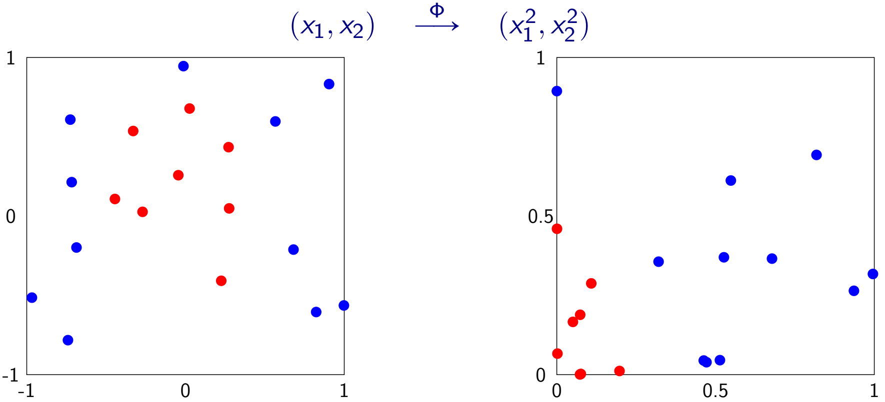
Data:



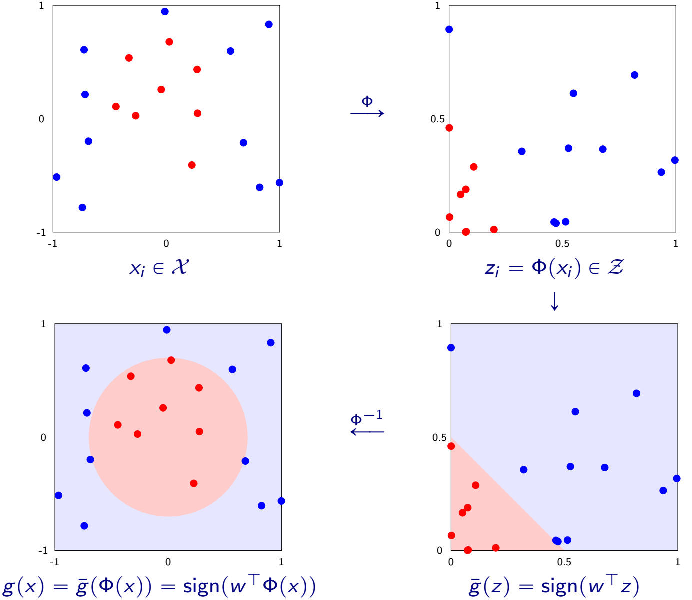
Hypothesis:



Non-linear feature transformation



Non-linear features with linear classifier



Feature transformation and linear classifier

$$\mathbf{x} = (x_0, x_1, \ldots, x_d) \quad \xrightarrow{\Phi} \quad \mathbf{z} = (z_0, z_1, \ldots, z_{\tilde{d}})$$

$$\mathbf{x}_1, \ldots, \mathbf{x}_n \quad \xrightarrow{\Phi} \quad (\mathbf{z}_1, \ldots, \mathbf{z}_n)$$

$$y_1, \ldots, y_n \quad \xrightarrow{\Phi} \quad y_1, \ldots, y_n$$

$$\text{No weights in } \mathcal{X} \quad \xrightarrow{\Phi} \quad \mathbf{w} = (w_0, w_1, \ldots, w_{\tilde{d}})$$

$$g(\mathbf{x}) \quad = \quad \mathrm{sign}(\mathbf{w}^\top \Phi(\mathbf{x}))$$

$$\text{Linear in } \mathbf{z} \text{ space } g(\mathbf{x}) \quad = \quad \mathrm{sign}(\mathbf{w}^\top \mathbf{z})$$

Transformation of Features

$\mathbf{x} = (x_0, x_1, \ldots, x_d) \xrightarrow{\Phi} \mathbf{z} = (z_0, z_1, \ldots, z_{\bar{d}})$ that is, $\mathbf{z} = \Phi(\mathbf{x})$

Polynomial of degree q (largest power) $\Phi(x) = \sum_{k=0}^{q} a_k x^k$

- Example: $\mathbf{x} = (1, x_1, x_2) \xrightarrow{\Phi} \mathbf{z} = (1, x_1, x_2, x_1^2, x_1 x_2, x_2^2)$

Final hypothesis with ERM

$$g(\mathbf{x}) = \underset{h \in \mathcal{H}}{\operatorname{argmin}} \widehat{R}_n(h) \quad g(\mathbf{x}) = \sum_{j=0}^{\bar{d}} w_j z_j$$

- classification $g(\Phi(\mathbf{x}), \mathbf{w}) = \operatorname{sign}(\mathbf{w}^\top \Phi(\mathbf{x}))$
- regression $g(\Phi(\mathbf{x}), \mathbf{w}) = \mathbf{w}^\top \Phi(\mathbf{x})$

Increased complexity of $\mathcal{H}$:

- Original VC dimension: $d_{VC} = d + 1$.
- New VC dimension: $d_{VC} \leqslant \bar{d} + 1$.

Learning with noisy data

- Data points ($\mathbf{x} \sim P(\mathbf{x})$):

$$(\mathbf{x}, y) \sim P(\mathbf{x}, y), \text{ that is } P(\mathbf{x}, y) = P(y|\mathbf{x})P(\mathbf{x})$$

- Target with noise (e.g. $\exists \mathbf{x}_a = \mathbf{x}_b : f(\mathbf{x}_a) \neq f(\mathbf{x}_b)$):

$$y = f(\mathbf{x}) + \mathit{noise} \text{ where } f(\mathbf{x}) = \mathbb{E}[y|\mathbf{x}] \text{ and } \mathbb{E}[\mathit{noise}|\mathbf{x}] = 0.$$

  Example: $y = w^\top x + N$ where $N \sim \mathcal{N}(0, \Sigma)$ is independent of $\mathbf{x}$.

- Deterministic target is a special case: $\mathit{noise} = 0$ and $P(y|\mathbf{x})$ is concentrated on the single point $f(\mathbf{x})$.

Learning problem

Choosing hypothesis class $\mathcal{H}$

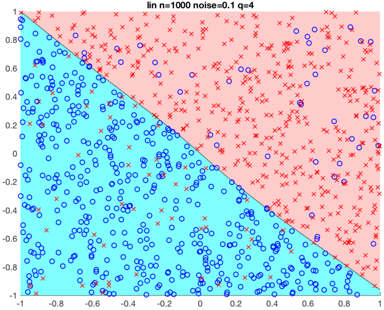
Learning $h(\mathbf{x}) = P(y|\mathbf{x})$.

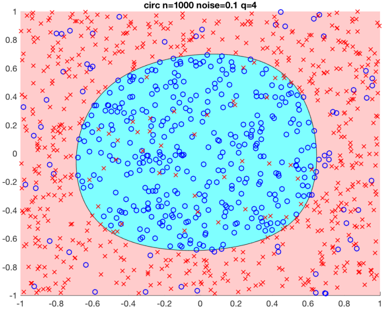
Learning with noisy data: generalisation, complexity, data size

Target : polynomial of degree q with noise $= 0.1$ and $n = 50$ samples

Target : polynomial of degree q with noise $= 0.1$ and $n = 1000$ samples

- Target A: $q = 1$ (linear)
- Target B: $q = 2$ (quadratic)





$$g(\mathbf{x}) = \operatorname{sign}(\mathbf{w}^\top \Phi(\mathbf{x}))$$

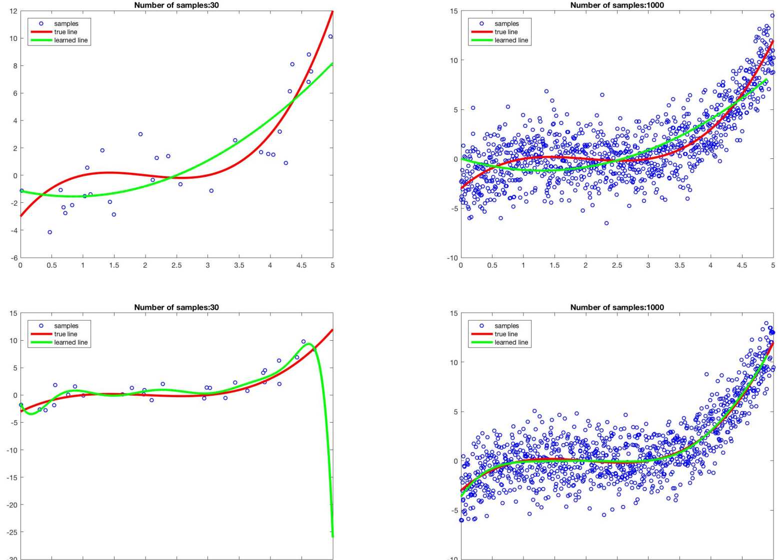
- Features:

Learning with noisy data: generalisation, complexity, data size

Fitting $f(x) = 0.5(x-1)*(x-2)*(x-3) + \mathcal{N}(0,4)$

degree 2

degree 10

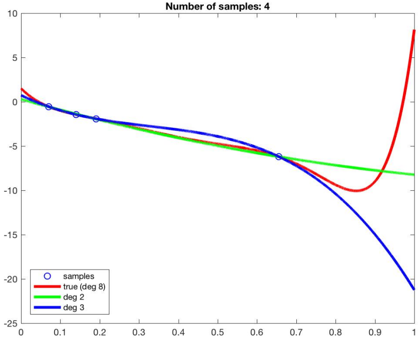


Learning with noisy data: generalisation, complexity, data size

- Target function: $f(x) = \sum_{k=0}^{q} a_k x^k + N$

  a polynomial of degree $q = 8$ on $[0, 1]$ (can fit any 9 data points)
- No noise: $N = 0$
- Data: $n = 4$ samples
- Hypothesis class: $\mathcal{H}_i$ of degree q_i ($q \sim$ complexity)

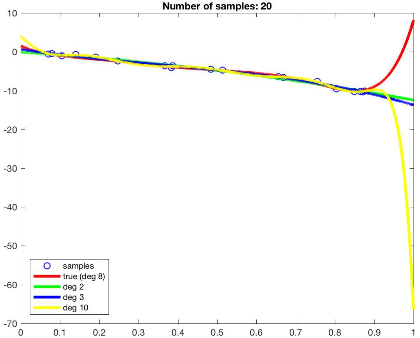
	$\mathcal{H}_1$ $q_1 = 2$	$\mathcal{H}_2$ $q_2 = 3$
$\widehat{R}_4$	0.0013	0
R	5.97	28.47



Learning with noisy data: generalisation, complexity, data size

- Target function: $f(x) = \sum_{k=0}^{q} a_k x^k + N$
  a polynomial of degree $q = 8$ on $[0, 1]$ (can fit any 9 data points)
- Noise: $N \sim \mathcal{N}(0, \sigma^2 = 0.25)$
- Data: $n = 20$ samples
- Hypothesis class: $\mathcal{H}_i$ of degree q_i ($q \sim$ complexity)

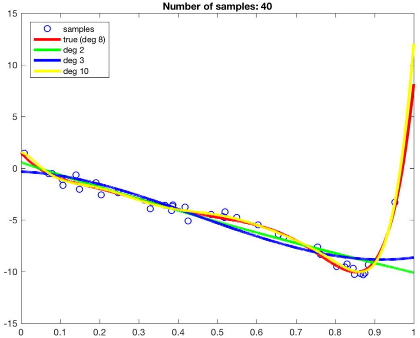
	$\mathcal{H}_1$	$\mathcal{H}_2$	$\mathcal{H}_3$
	$q_1 = 2$	$q_2 = 3$	$q_3 = 10$
$\widehat{R}_{20}$	0.12	0.084	0.022
R	10.66	12.28	79.72



Learning with noisy data: generalisation, complexity, data size

- Target function: $f(x) = \sum_{k=0}^{q} a_k x^k + N$
  a polynomial of degree $q = 8$ on $[0, 1]$ (can fit any 9 data points)
- Noise: $N \sim \mathcal{N}(0, \sigma^2 = 0.25)$
- Data: $n = 40$ samples
- Hypothesis class: $\mathcal{H}_i$ of degree q_i ($q \sim$ complexity)

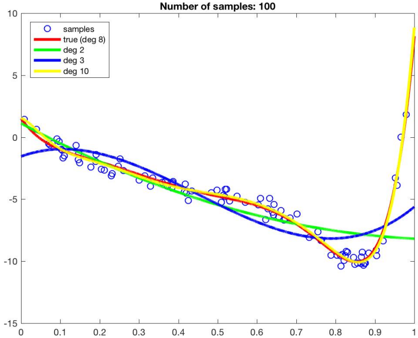
	$\mathcal{H}_1$	$\mathcal{H}_2$	$\mathcal{H}_3$
	$q_1 = 2$	$q_2 = 3$	$q_3 = 10$
$\widehat{R}_{40}$	1.56	1.47	0.12
R	7.54	6.30	0.25



Learning with noisy data: generalisation, complexity, data size

- Target function: $f(x) = \sum_{k=0}^{q} a_k x^k + N$
  a polynomial of degree $q = 8$ on $[0, 1]$ (can fit any 9 data points)
- Noise: $N \sim \mathcal{N}(0, \sigma^2 = 0.25)$
- Data: $n = 100$ samples
- Hypothesis class: $\mathcal{H}_i$ of degree q_i ($q \sim$ complexity)

	$\mathcal{H}_1$	$\mathcal{H}_2$	$\mathcal{H}_3$
	$q_1 = 2$	$q_2 = 3$	$q_3 = 10$
$\widehat{R}_{100}$	3.29	2.76	0.18
R	5.88	4.33	0.03

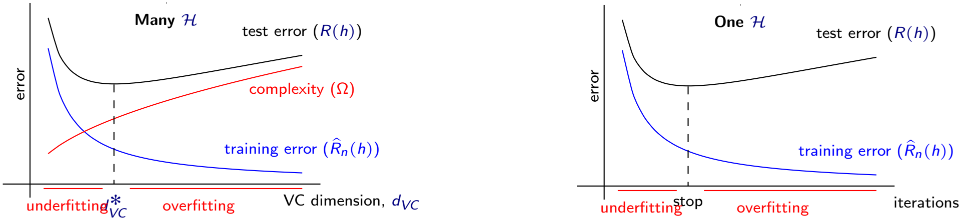


Learning with noisy data: overfitting

Overfitting

Occurs when $\widehat{R}_n(h) \downarrow \quad R(h) \uparrow$, moving away from the target.

Fitting to noise instead of the underlying target function/distribution.



Remember n matters too!

Noise types: stochastic ($N \sim \mathcal{N}(0, \sigma^2)$), deterministic (complexity)

deterministic noise	$\uparrow$	overfitting	$\uparrow$
stochastic noise	$\uparrow$	overfitting	$\uparrow$
number of data points	$\uparrow$	overfitting	$\downarrow$

Learning with noisy data: deterministic and stochastic noise

Target function: $y = f(\mathbf{x}) + N$ with $N \sim \mathcal{N}(0, \Sigma)$

Deterministic noise–pointwise bias: $f(\mathbf{x}) - h^*(\mathbf{x})$

- h^* is the best approximation to f in $\mathcal{H}$

$$h^* = \operatorname*{argmin}_{h \in \mathcal{H}} \mathbb{E}_x \left[\ell(h(\mathbf{x}), f(\mathbf{x}))\right]$$

- deterministic noise depends on $\mathcal{H}$ and f
- fixed for a given $\mathbf{x}$

Bias-variance decomposition :

$$\mathbb{E}_{\mathcal{D},N}\left[\left(g^{(\mathcal{D})}(\mathbf{x}) - y\right)^2\right] = \mathbb{E}_{\mathcal{D},N}\left[\left(g^{(\mathcal{D})}(\mathbf{x}) - f(\mathbf{x}) - N(\mathbf{x})\right)^2\right]$$

$$= \underbrace{\mathbb{E}_{\mathcal{D}}\left[\left(g^{(\mathcal{D})}(\mathbf{x}) - \bar{g}(\mathbf{x})\right)^2\right)\right]}_{\text{var}} + \underbrace{\mathbb{E}_{\mathcal{D}}\left[\left(\bar{g}(\mathbf{x}) - f(\mathbf{x})\right)^2\right)\right]}_{\substack{\text{bias} \\ \text{deterministic noise}}} + \underbrace{\mathbb{E}_{N}\left[\left(N(\mathbf{x})\right)^2\right]}_{\substack{\sigma^2 \\ \text{stochastic noise}}}$$

Learning with noisy data: Improving features

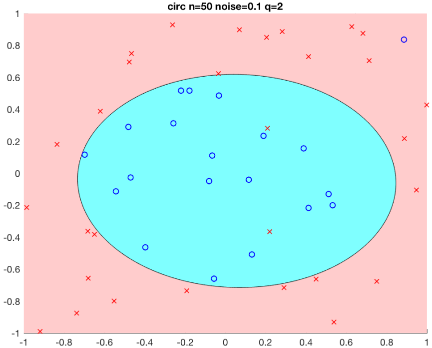
$$\mathbf{x} = (1, x_1, x_2) \to \mathcal{H}_1$$

$$\mathbf{z} = (1, x_1, x_2, x_1^2, x_1 x_2, x_2^2) \to \mathcal{H}_2$$

But why not $\mathbf{z} = (1, x_1^2, x_2^2) \to \mathcal{H}_3$

or even better $\mathbf{z} = (1, x_1^2 + x_2^2) \to \mathcal{H}_4$

or simply $\mathbf{z} = (x_1^2 + x_2^2 - 0.49) \to \mathcal{H}_5$



Data snooping

Incorporating prior knowledge is good, but

Looking at the data before choosing the model may be risky!

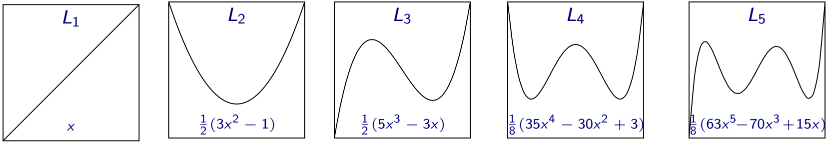
Better features: Legendre polynomials

$$y = f(x) + N = \sum_{q=1}^{Q_f} a_q L_q(x) + N$$

- $L_q(x)$: Legendre polynomials, form an **orthogonal basis** for piecewise smooth functions on $[-1, 1]$.

$$L_q(x) = \frac{1}{2^q q!} \frac{d^q}{dx^q}(x^2 - 1)^q \qquad \text{with} \qquad \mathbb{E}_x\left[L_q^2(x)\right] = \frac{2}{2q+1}$$

- $\frac{d^q}{dx^q}$: q-order derivative.
- a_q standard normal, normalized such that $\mathbb{E}_{a,x}\left[f(x)^2\right] = 1$.
- N zero-mean Gaussian noise, with variance σ^2
- Hypothesis class of degree Q, $\mathcal{H}_Q = \left\{\sum_{q=0}^{Q} w_q L_q(x)\right\}$



Constraining the hypothesis class $\longrightarrow$ regularisation!

Learning with noisy data: Which $\mathcal{H} \sim \Phi(\mathbf{x})$ should we choose?

Given many hypotheses classes $\mathcal{H}_1, \mathcal{H}_2, \ldots$

Which $\mathcal{H}_i$ should we choose g from?

Use data to select g!

Structural Risk Minimization

Regularisation

Validation