Machine Learning I: Supervised Methods

B. Keith Jenkins

Announcements

- Slido event code: 5499248
  - slido.com
- Posted on D2L:
  - Class-participation grading criteria
- Announced on piazza:
  - Instructions to register regular recurring lecture-time conflicts
  - Homework file submission - please follow stated guidelines; otherwise you will lose points (HW4 and after)
- Homework 4 is due Friday
- No new homework will be posted this Friday
  - HW5 will be posted Friday 2/23
- Midterm exam is Wed., 3/6/2024
  - Vote for materials allowed: watch for piazza post with poll
  - 1 hour 45 min. exam
- No class on Monday 2/19 - Presidents' Day

Reading

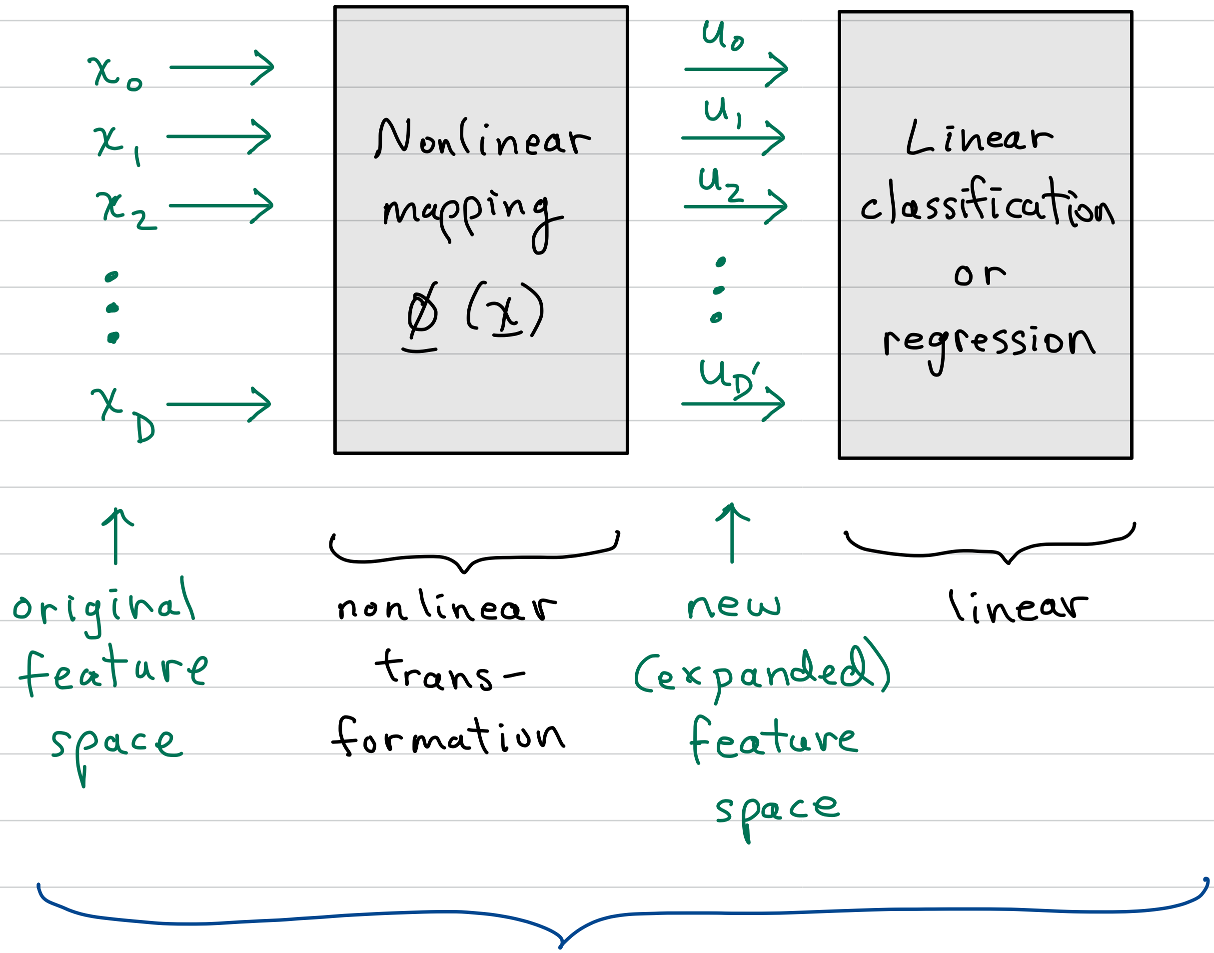
- Bishop 6.0-6.2, App. E
  - Lagrangian optimization; kernels

Machine Learning I: Supervised Methods

B. Keith Jenkins

Today's lecture

- Nonlinear classification and regression
  - Learning
- Complexity in supervised learning
  - Degrees of freedom
  - Constraints
  - Example



Classification:

$g(\underline{u}) = \underline{w}_k'^T \underline{u}$

→ Linear in $\underline{u}$

→ Linear in $\underline{w}_k'$

$\underline{\Phi}$ machine (Nilsson) – $\underline{\phi}(\underline{x})$ are polynomials in $\underline{x}$ (originally)

Also called (for general nonlinear functions of $\underline{x}$):

$\underline{\phi}(\underline{x})$: "nonlinear transformation"

or "basis-set expansion"

$g(\underline{u})$ or $\hat{f}(\underline{u})$: "generalized linear (discriminant or regression) functions"

Learning nonlinear (in $\underline{x}$) models

1. Use nonlinear transformation $\underline{\phi}(\underline{x})$ to map $\underline{x}$ into $\underline{u}$-space. (pre-processing)
2. ⊛ Use any linear learning algorithm to find $\hat{\underline{w}}'_k$ or $\hat{\underline{w}}'$.
3. Optionally, map:

   (i) decision boundaries (and regions)
   or (ii) regression function $\hat{f}(\underline{u})$

   back into original feature space ($\underline{x}$-space).
4. How to make predictions on unknowns $\underline{x}$, given $\hat{\underline{w}}'$ or $\hat{\underline{w}}'_k$?

   a. $\underline{u} = \underline{\phi}(\underline{x})$

   ⊛ b. Calculate $g_k(\underline{u}) = \hat{\underline{w}}'^T_k \underline{u}$ then apply decision rule (classification)

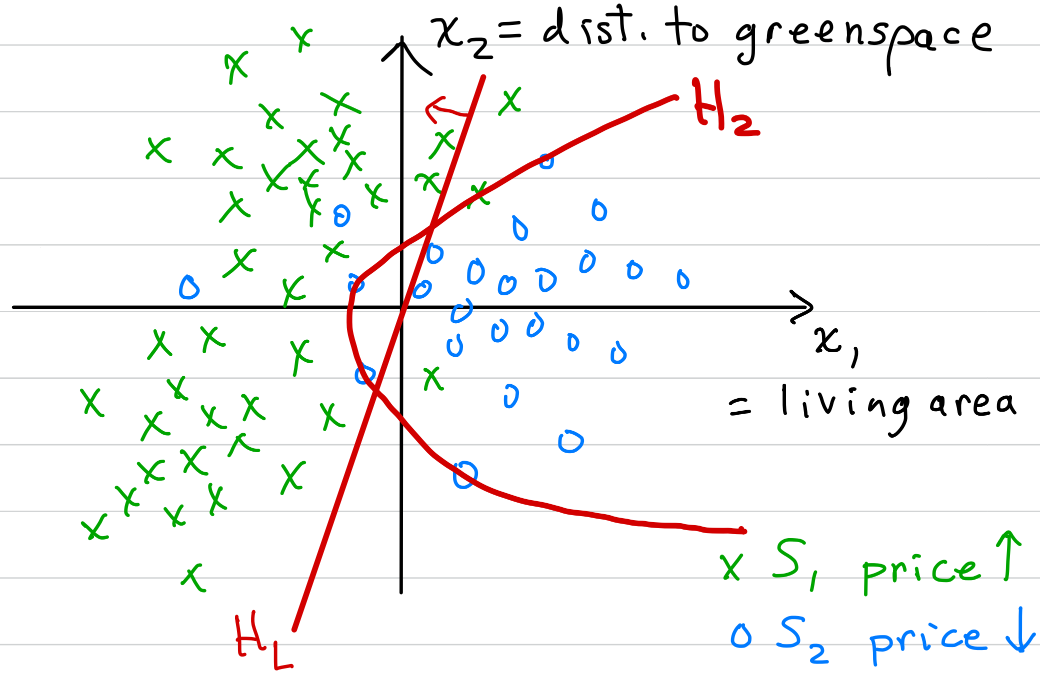
   ⊛ $\underline{\underline{\text{or}}}$ Calculate $\hat{y} = \hat{f}(\underline{u}) = \underline{w}'^T \underline{u}$ (regression)

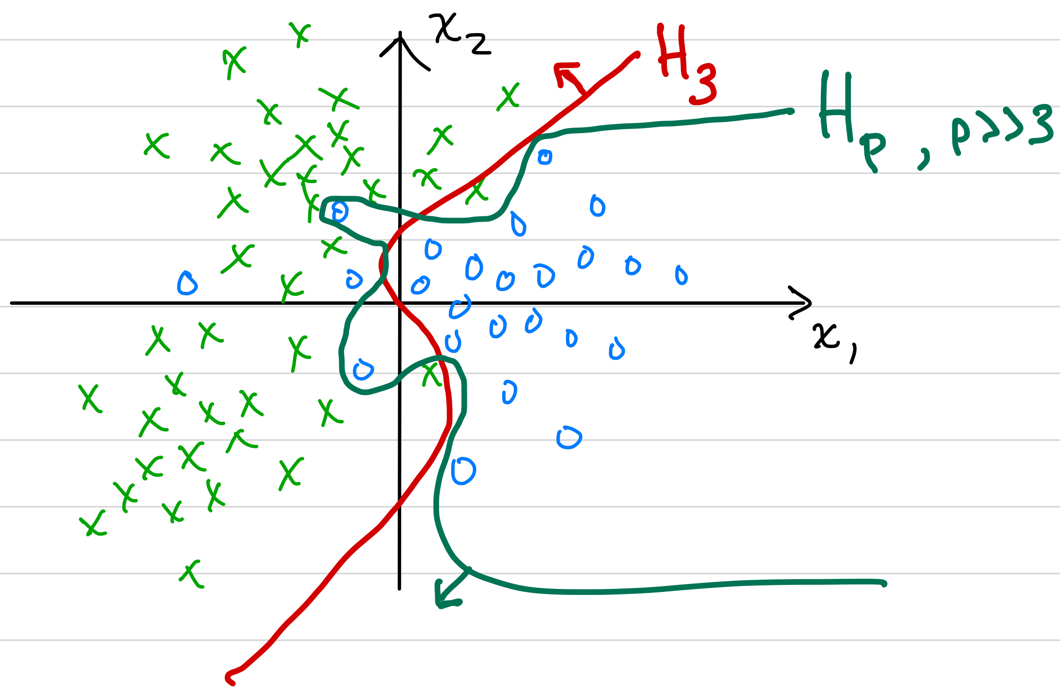
⊛ Same as linear case, except variable change $\underline{x} \to \underline{u}$, $\hat{\underline{w}} \to \hat{\underline{w}}'$

We will give other examples of $\underline{\phi}$ mappings later [with kernels SVM's].

Complexity of Learning by Machine

Nonlinear transformation of $\underline{x}$ before learning	# scalar weight variables to learn
Linear	$D+1$
2^{nd} order polynomial	$\frac{1}{2}(D^2+3D)+1 = O(D^2)$
3^{rd} order polynomial	$O(D^3)$
p^{th} order polynomial	$O(D^p)$





$\Longrightarrow$ Generalization depends on how flexible the decision boundary can be!

S

Complexity of Learning – degrees of freedom (d.o.f.) and constraints (N_c)

Consider a system of linear equations and unknowns: (augmented notation)

$$\underset{[N \times (D+1)]}{\underline{\underline{X}}} \; \underset{[D+1]}{\underline{w}} = \underset{[N]}{\underline{b}} \quad \text{(orig. feat. space)}$$

or

$$\underset{[N \times (D'+1)]}{\underline{\underline{\Phi}}} \; \underset{[D'+1]}{\underline{w}'} = \underset{[N]}{\underline{b}'} \quad \text{(expanded feature space)}$$

# Eqns.	# unknowns (variables w_j or w_j')	
N	$D+1$	(orig. feat. space)
N	$D'+1$	(expanded feat. space)
$\approx N_c$	d.o.f.	

We can define <u>degrees of freedom</u> (d.o.f.) in a learning problem as the number of variables that the learning algorithm can adjust independently.

<u>Ex</u>: $g(\underline{x}) = \underline{w}^T \underline{x}$ (augm. notation) (2-class)

$$\text{d.o.f.} = D+1$$

<u>Constraints</u> constrain choices of these variables.

<u>Ex</u>: $\underline{\underline{X}}\,\underline{w} = \underline{b}$ is a set of N constraints on $\underline{w}$.

Let $N_c = \#$ of constraints.

Complexity: N_c and d.o.f.

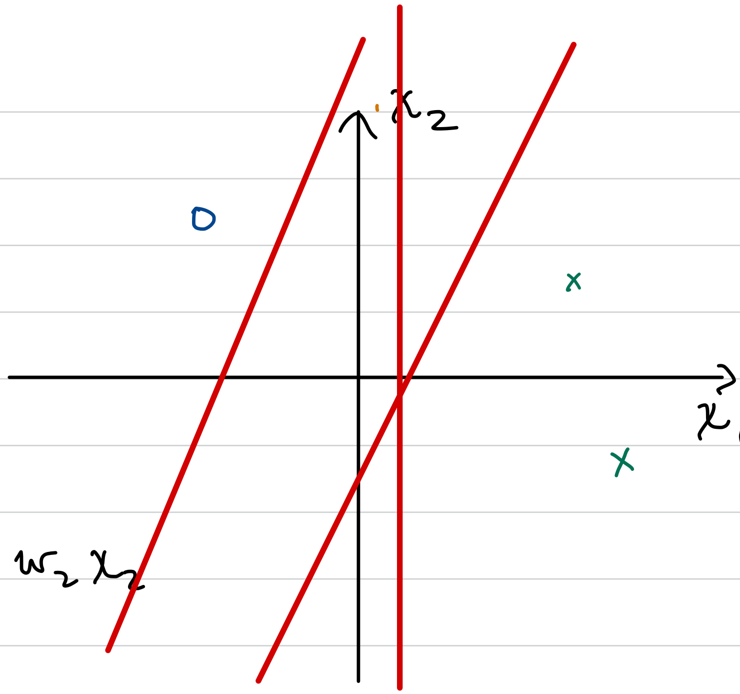
Consider: $\underline{\underline{X}}\,\underline{w} = \underline{b}$ (original feat. spc.)

$\underline{\underline{\Phi}}\,\underline{w}' = \underline{b}'$ (expanded feat. spc.)

Feature space	$N_c \leftrightarrow$ d.o.f.	system of eqns.	Comments
orig.	$N = D+1$	exactly determined	if $\underline{\underline{X}}$ or $\underline{\underline{\Phi}}$ is full rank
expanded	$N = D'+1$	" "	
(more generally, N_c = d.o.f.)			
orig.	$N > D+1$	over determined	if $\underline{\underline{X}}$ or $\underline{\underline{\Phi}}$ has full column rank.
expanded	$N > D'+1$	" "	
(more generally, $N_c >$ d.o.f.)			
orig.	$N < D+1$	under determined	
expanded	$N < D'+1$	" "	
(more generally, $N_c <$ d.o.f.)			

Ex :

Linear model for
2-class problem
$N = 3$



$\times S_1$ (house price ↑)
$\circ S_2$ (house price ↓)

wide variety of decision boundaries is possible.

$$g(\underline{x}) = w_0 + w_1 x_1 + w_2 x_2$$

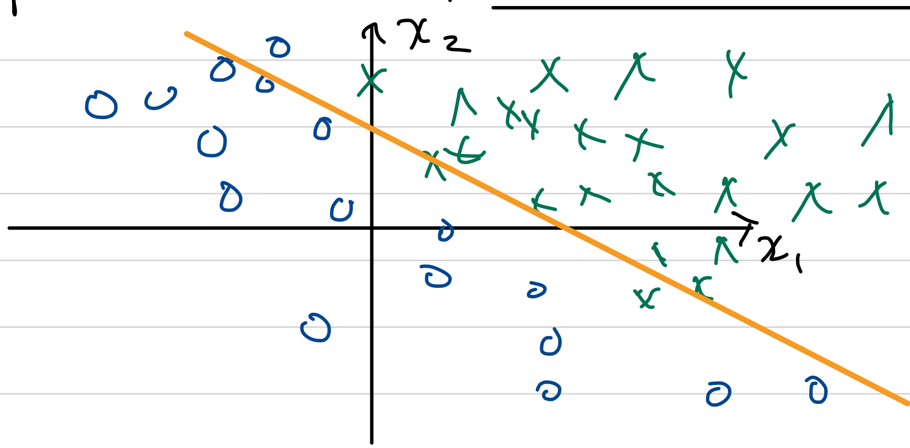
$$\underline{\underline{X}}\,\underline{w} = \underline{b}$$

$$N_c = \#\text{eqns.} = 3$$
$$\text{d.o.f.} = \# w_j = 3$$

$\Rightarrow$ exactly determined if $\underline{\underline{X}}$ is of full rank.

$\rightarrow$ In ML, we prefer to learn in overdetermined realm.

e.g.



<u>General rule of thumb:</u>

Typically we want $N_c > (3\text{-}10) \times \text{d.o.f.}$

e.g., $N > (3\text{-}10) \times (D+1)$ for linear discr. fcn case, orig. feat. space, given above

(Empirical / numerical result.)

What if N_c < $(3\text{-}10) \times$ d.o.f.?

$\Rightarrow$ Risk of overfitting

Possible approaches:

1. Increase N_c by increasing N = # data points
   - Collect more data
   - Synthetically generate more data

2. Increase N_c by adding other constraints
   - Use prior knowledge in some way
   - Restrict choices of optimal $\underline{w}$

$\longrightarrow$ 3. Decrease d.o.f.
   - Reduce D or D' (e.g., feature selection or transformation)
   - Reduce amount of nonlinearity in the model

$\uparrow$

Today

Example of d.o.f. and N_c

Consider a 1D regression problem [Bishop, Sec. 1.1]

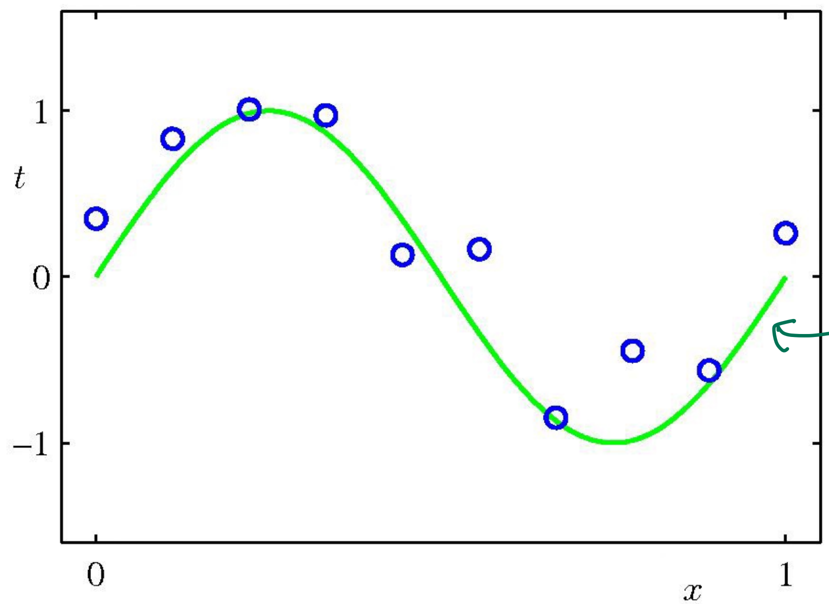
Let the known output values y_i in the dataset come from a target fcn:

$$y = \sin(2\pi x) + \text{noise}$$

thus: $\mathcal{D} = \{(x_i, y_i),\ 0 \leq x_i \leq 1$ (uniformly spaced on $[0,1]$),
$y_i = \sin(2\pi x_i) + n_i,\ n_i \sim N(0, \sigma^2),\ i = 1, 2, \cdots, N\}$

Let $N = 10$.

y



Target function (without noise)

only one feature

x

We will use a set of (mostly nonlinear) models – polynomials in x:

$$\hat{f}_d(x) = \sum_{i=0}^{d} w_i' x^i, \quad d \text{ is a parameter we specify.}$$

Let $\underline{u} = \begin{bmatrix} x^0 \\ x^1 \\ x^2 \\ \vdots \\ x^d \end{bmatrix}$, $\underline{w}'$ is defined above.

Ⓢ

Expanded f.s. dimensionality = ? $d+1$

Original f.s. dimensionality = ? 1 feature x

$\Rightarrow$ dim. 1 (non-augmented)

dim. 2 (augmented)

d.o.f. = ? $d+1$ scalar w_j' values.

N_c = ? N data points

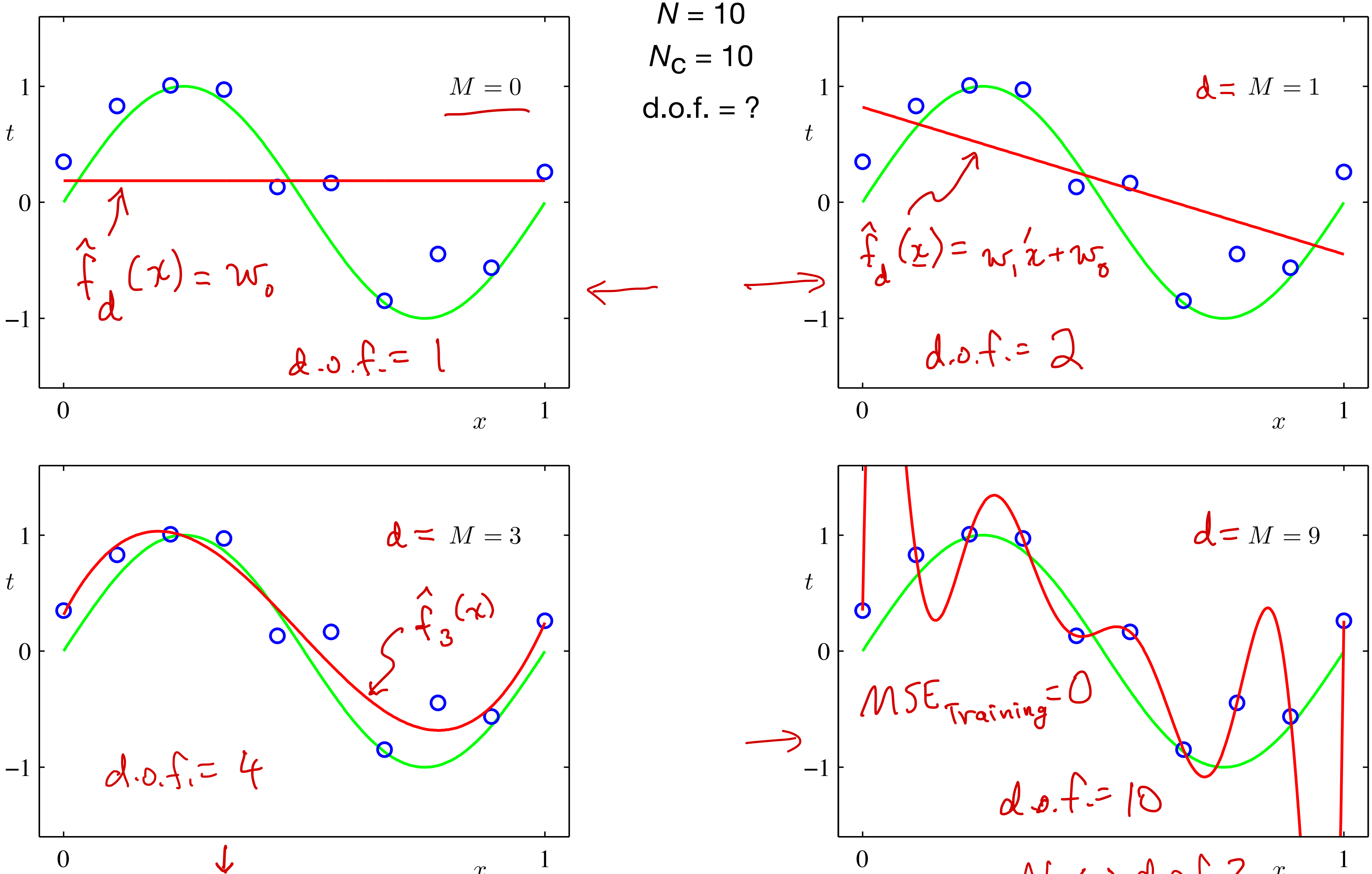
Use MSE criterion $\longrightarrow$ least-squares regression.

Regression complexity example: varying *d*

$N = 10$

$N_c = 10$

d.o.f. = ?



$N_c \leftrightarrow$ d.o.f.?

N_c: 10 — d.o.f.: 4

$N_c \leftrightarrow$ d.o.f.?

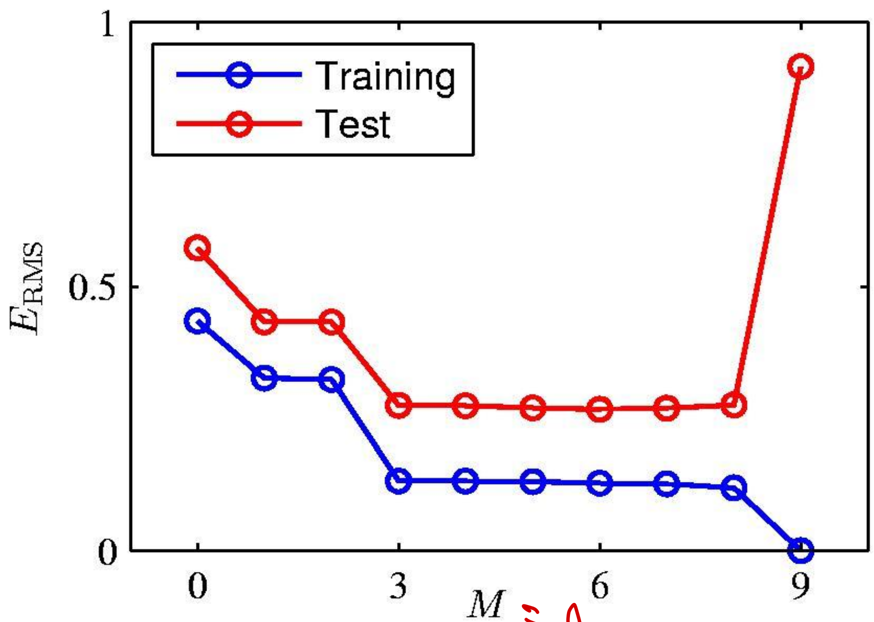
N_c: 10 — d.o.f.: 10

Bishop, *Pattern Recognition and Machine Learning*, Fig. 1.4

Regression complexity example: varying *d*

$N = 10$

$N_C = 10$



$= d$

d.o.f. = d+1

Polynomial Coefficients

	$M = 0$	$M = 1$	$M = 3$	$M = 9$
$\hat{w}_0$ → $w_0^\star$	0.19	0.82	0.31	0.35
$\hat{w}_1$ → $w_1^\star$		-1.27	7.99	232.37
⋮ $w_2^\star$			-25.43	-5321.83
$w_3^\star$			17.37	48568.31
$w_4^\star$				-231639.30
$w_5^\star$				640042.26
$w_6^\star$				-1061800.52
$w_7^\star$				1042400.18
$w_8^\star$				-557682.99
$\hat{w}_9$ → $w_9^\star$				125201.43

Bishop, *Pattern Recognition and Machine Learning*, Fig. 1.5 and Table 1.1