

LEARNING FROM DATA

Machine Learning course - recorded at a live broadcast from Caltech

HOMEWORK

The course has **8 homework sets plus a Final**, according to the [schedule below](#). The Final carries twice the weight of a homework.

- The questions are multiple-choice. This doesn't mean simplistic; some questions necessitate running a full experiment.
- Some questions are theoretical (mathematical, conceptual), and others are experimental (programming, data). The questions vary from easy to hard.
- Familiarity with some programming language or platform will help in the homework, e.g., packages for basic matrix operations and quadratic programming are needed.
- [The discussion forum](#) has numerous threads about homework and Final questions.

[Home](#)

[The lectures](#)

[\[Homework\]](#)

[Textbook](#)

[Forum](#)

[The instructor](#)

[Contact](#)



Homework Schedule

Take the course at your own pace. A 'week' below can be spread out to a longer period, e.g., 10 days or two weeks, to fit your schedule.

- **Week 1:** Do [Homework 1](#) after watching Lectures 1 and 2. Check [Solution key 1](#) after you finish the homework.
- **Week 2:** Do [Homework 2](#) after watching Lectures 3 and 4. Check [Solution key 2](#) after you finish the homework.
- **Week 3:** Do [Homework 3](#) after watching Lectures 5 and 6. Check [Solution key 3](#) after you finish the homework.
- **Week 4:** Do [Homework 4](#) after watching Lectures 7 and 8. Check [Solution key 4](#) after you finish the homework.
- **Week 5:** Do [Homework 5](#) after watching Lectures 9 and 10. Check [Solution key 5](#) after you finish the homework.
- **Week 6:** Do [Homework 6](#) after watching Lectures 11 and 12. Check [Solution key 6](#) after you finish the homework.
- **Week 7:** Do [Homework 7](#) after watching Lectures 13 and 14. Check [Solution key 7](#) after you finish the homework.
- **Week 8:** Do [Homework 8](#) after watching Lectures 15 and 16. Check [Solution key 8](#) after you finish the homework.
- **Week 9:** Start on the [Final](#) after watching Lectures 17 and 18. Consult the [Machine Learning Video Library](#) as needed.
- **Week 10:** Finish the [Final](#) then check the [Final solution key](#). Congratulations! You have completed the course.



©2015 [California Institute of Technology](#). All rights reserved.

Outline of the Course

1. The Learning Problem (*April 3*)
2. Is Learning Feasible? (*April 5*)
3. The Linear Model I (*April 10*)
4. Error and Noise (*April 12*)
5. Training versus Testing (*April 17*)
6. Theory of Generalization (*April 19*)
7. The VC Dimension (*April 24*)
8. Bias-Variance Tradeoff (*April 26*)
9. The Linear Model II (*May 1*)
10. Neural Networks (*May 3*)

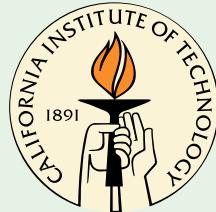
11. Overfitting (*May 8*)
12. Regularization (*May 10*)
13. Validation (*May 15*)
14. Support Vector Machines (*May 17*)
15. Kernel Methods (*May 22*)
16. Radial Basis Functions (*May 24*)
17. Three Learning Principles (*May 29*)
18. Epilogue (*May 31*)

- theory; mathematical
- technique; practical
- analysis; conceptual

Learning From Data

Yaser S. Abu-Mostafa
California Institute of Technology

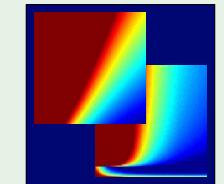
Lecture 1: The Learning Problem



Sponsored by Caltech's Provost Office, E&AS Division, and IST



Tuesday, April 3, 2012



The learning problem - Outline

- Example of machine learning
- Components of Learning
- A simple model
- Types of learning
- Puzzle

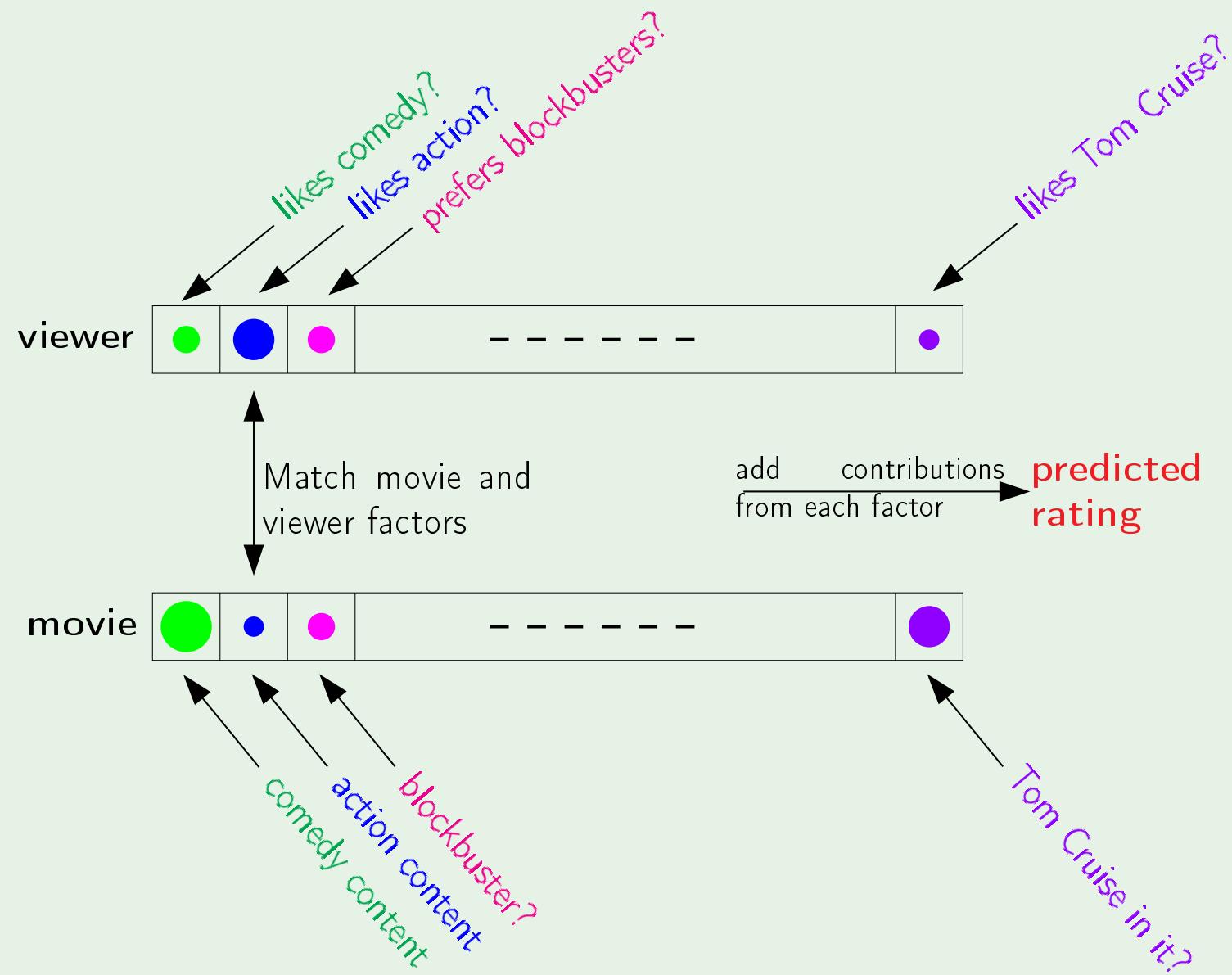
Example: Predicting how a viewer will rate a movie

10% improvement = 1 million dollar prize

The essence of machine learning:

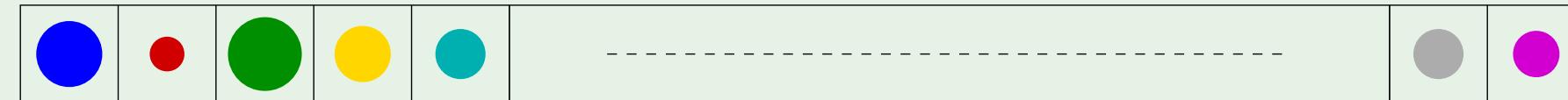
- A pattern exists.
- We cannot pin it down mathematically.
- We have data on it.

Movie rating - a solution

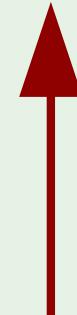
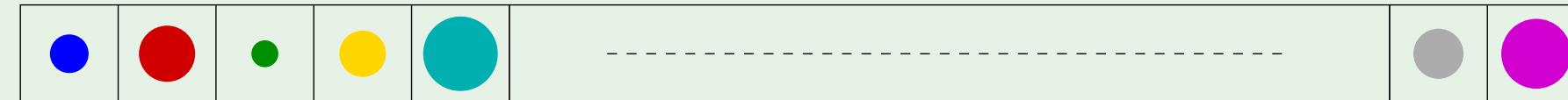


The learning approach

v i e w e r



m o v i e



LEARNING

rating

Components of learning

Metaphor: Credit approval

Applicant information:

age	23 years
gender	male
annual salary	\$30,000
years in residence	1 year
years in job	1 year
current debt	\$15,000
...	...

Approve credit?

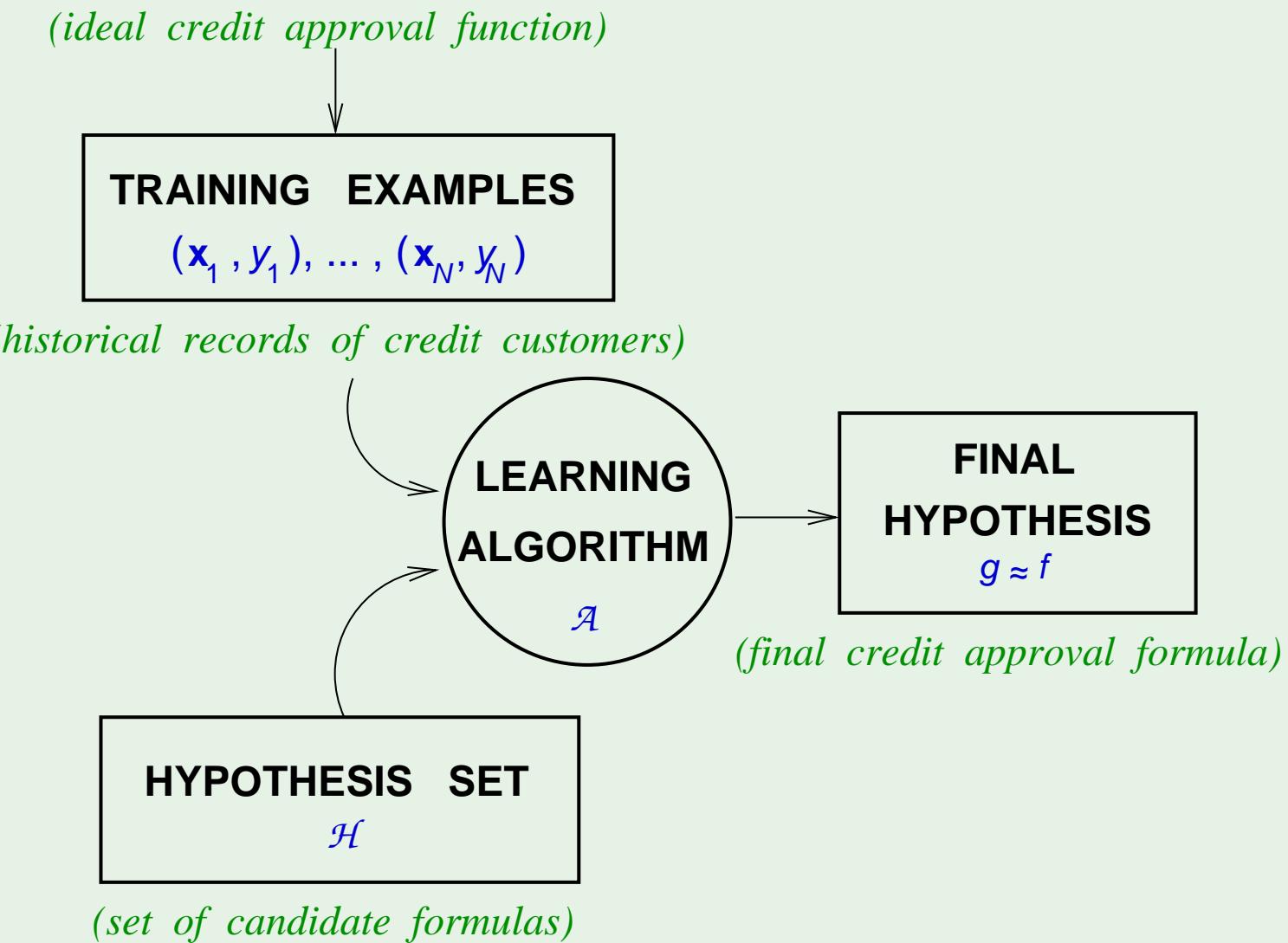
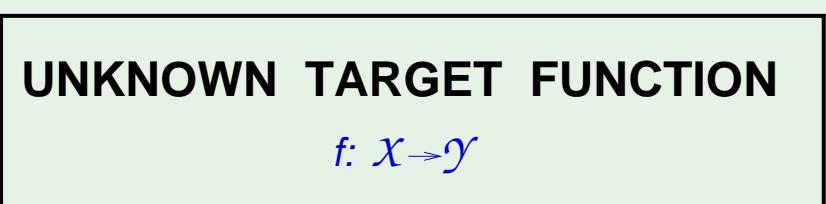
Components of learning

Formalization:

- Input: \mathbf{x} (*customer application*)
- Output: y (*good/bad customer?*)
- Target function: $f : \mathcal{X} \rightarrow \mathcal{Y}$ (*ideal credit approval formula*)
- Data: $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_N, y_N)$ (*historical records*)



- Hypothesis: $g : \mathcal{X} \rightarrow \mathcal{Y}$ (*formula to be used*)



Solution components

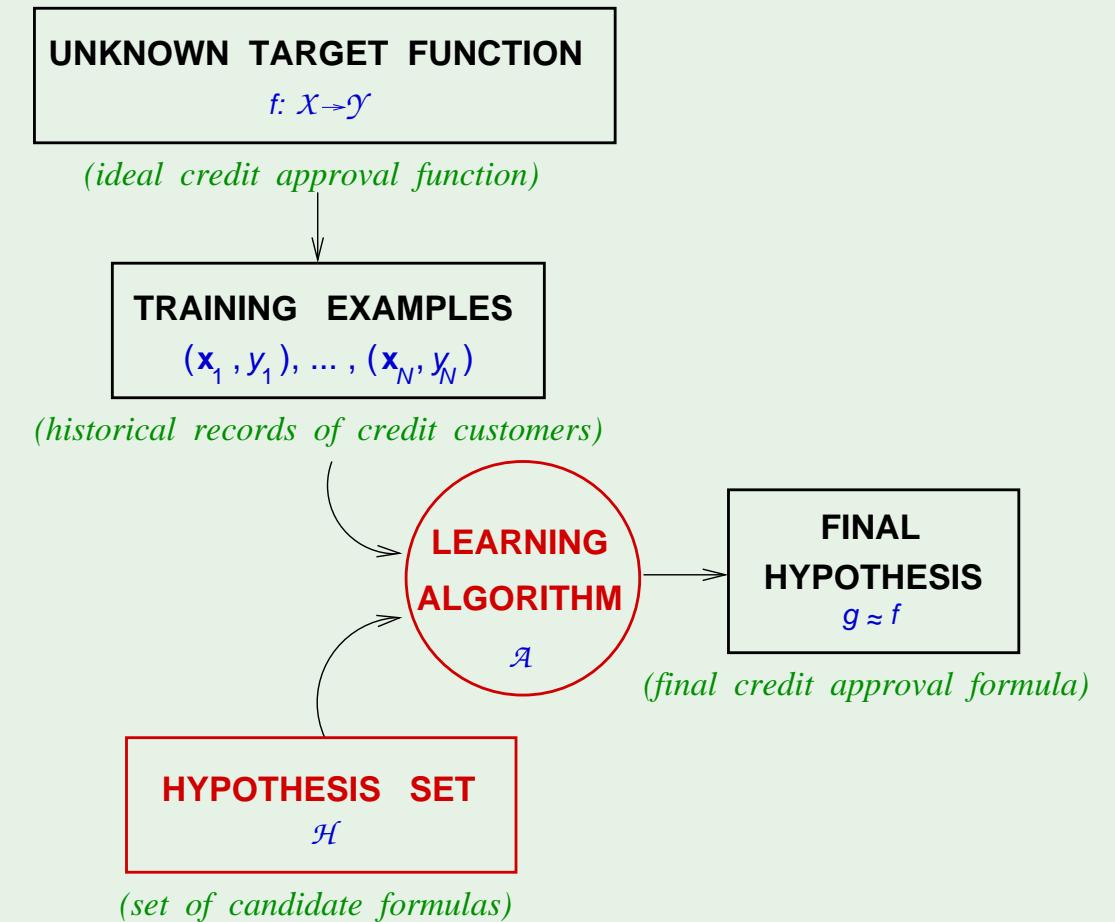
The 2 solution components of the learning problem:

- The Hypothesis Set

$$\mathcal{H} = \{h\} \quad g \in \mathcal{H}$$

- The Learning Algorithm

Together, they are referred to as the *learning model*.



A simple hypothesis set - the ‘perceptron’

For input $\mathbf{x} = (x_1, \dots, x_d)$ ‘attributes of a customer’

Approve credit if $\sum_{i=1}^d w_i x_i > \text{threshold}$,

Deny credit if $\sum_{i=1}^d w_i x_i < \text{threshold}$.

This linear formula $h \in \mathcal{H}$ can be written as

$$h(\mathbf{x}) = \text{sign} \left(\left(\sum_{i=1}^d \color{red}{w_i} x_i \right) - \text{threshold} \right)$$

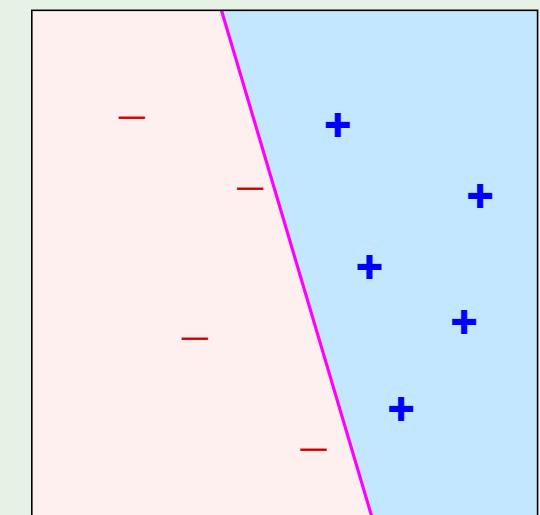
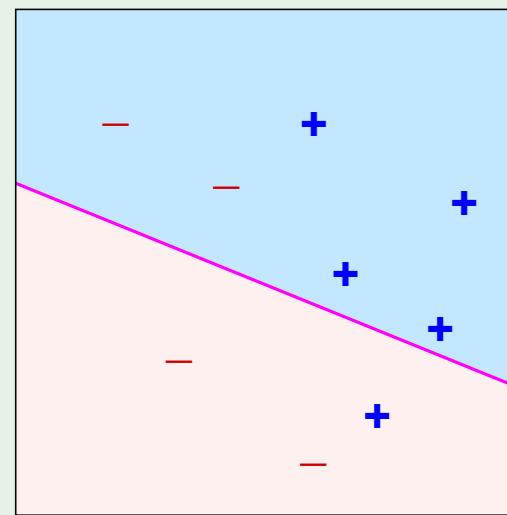
$$h(\mathbf{x}) = \text{sign} \left(\left(\sum_{i=1}^d w_i x_i \right) + w_0 \right)$$

Introduce an artificial coordinate $x_0 = 1$:

$$h(\mathbf{x}) = \text{sign} \left(\sum_{i=0}^d w_i x_i \right)$$

In vector form, the perceptron implements

$$h(\mathbf{x}) = \text{sign}(\mathbf{w}^\top \mathbf{x})$$



'linearly separable' data

A simple learning algorithm - PLA

The perceptron implements

$$h(\mathbf{x}) = \text{sign}(\mathbf{w}^\top \mathbf{x})$$

Given the training set:

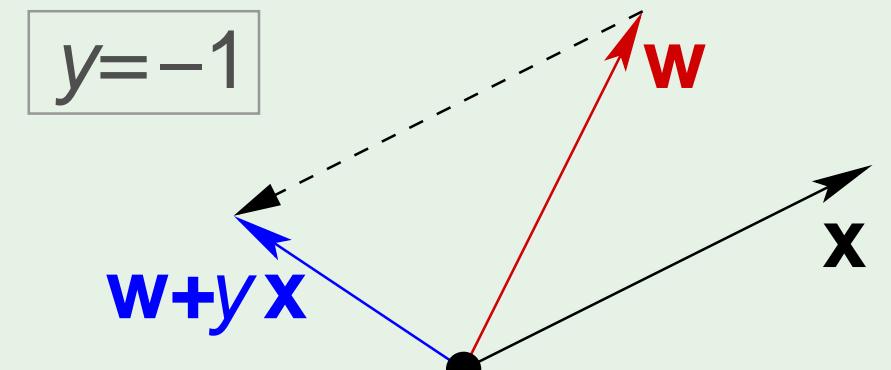
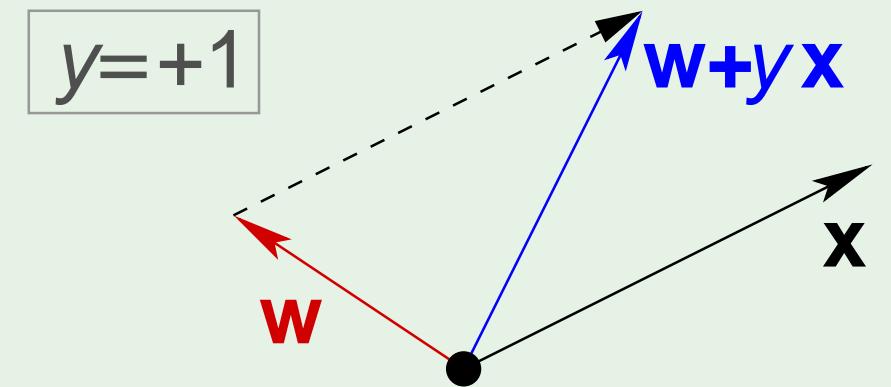
$$(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_N, y_N)$$

pick a **misclassified** point:

$$\text{sign}(\mathbf{w}^\top \mathbf{x}_n) \neq y_n$$

and update the weight vector:

$$\mathbf{w} \leftarrow \mathbf{w} + y_n \mathbf{x}_n$$



Iterations of PLA

- One iteration of the PLA:

$$\mathbf{w} \leftarrow \mathbf{w} + y\mathbf{x}$$

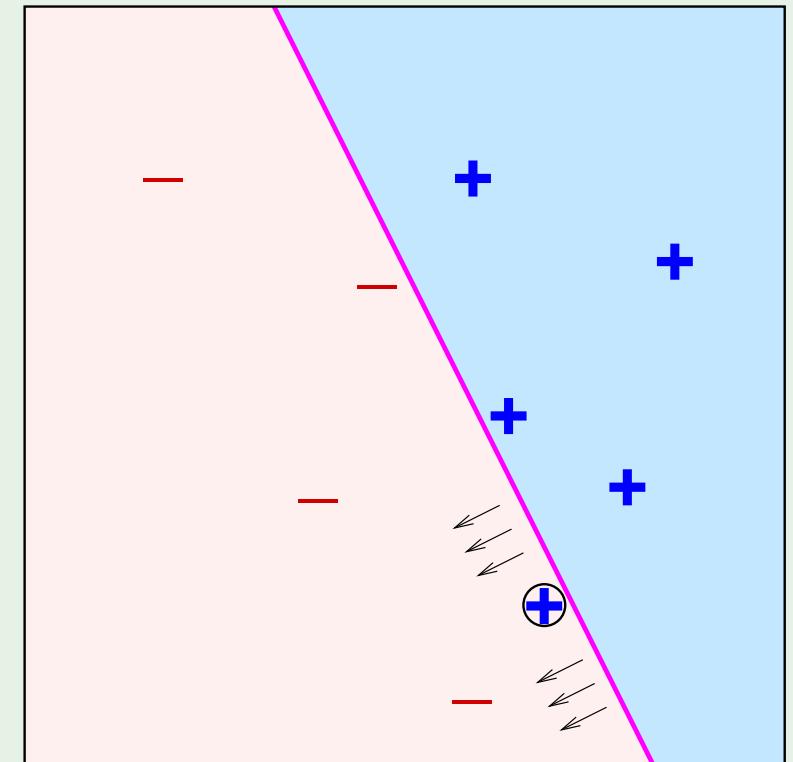
where (\mathbf{x}, y) is a misclassified training point.

- At iteration $t = 1, 2, 3, \dots$, pick a misclassified point from

$$(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_N, y_N)$$

and run a PLA iteration on it.

- That's it!



The learning problem - Outline

- Example of machine learning
- Components of learning
- A simple model
- Types of learning
- Puzzle

Basic premise of learning

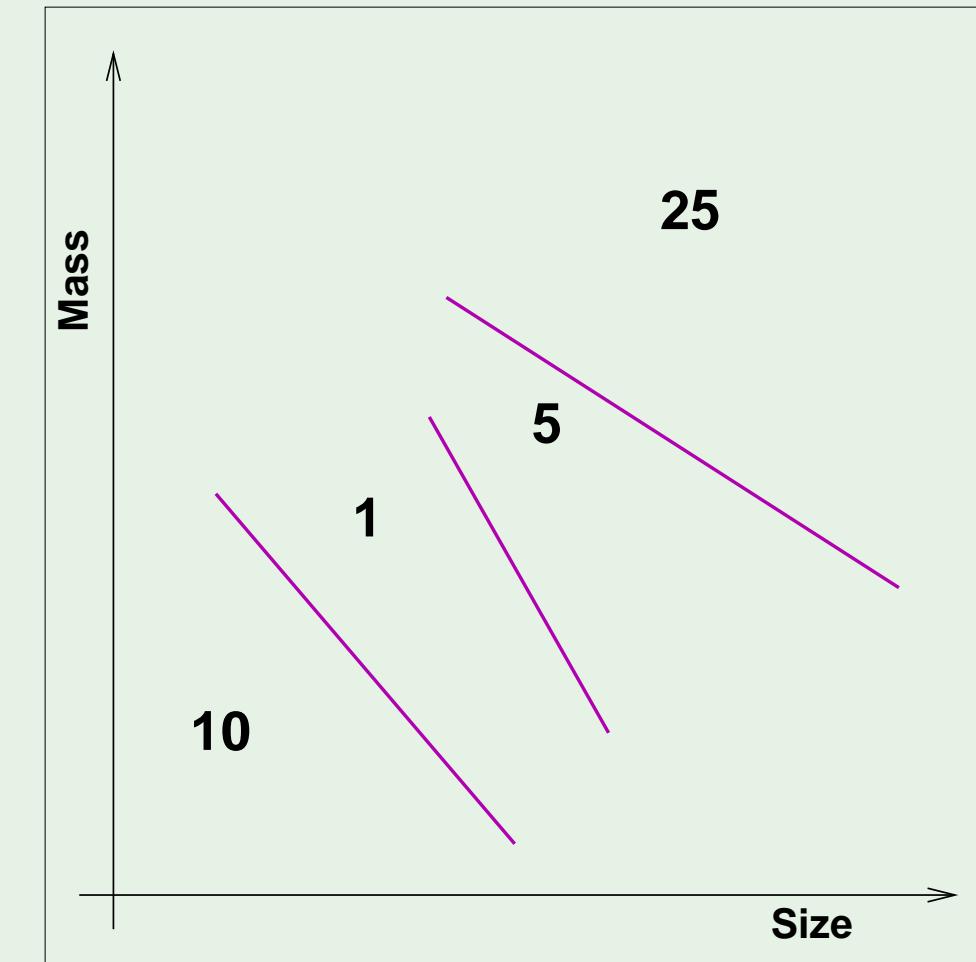
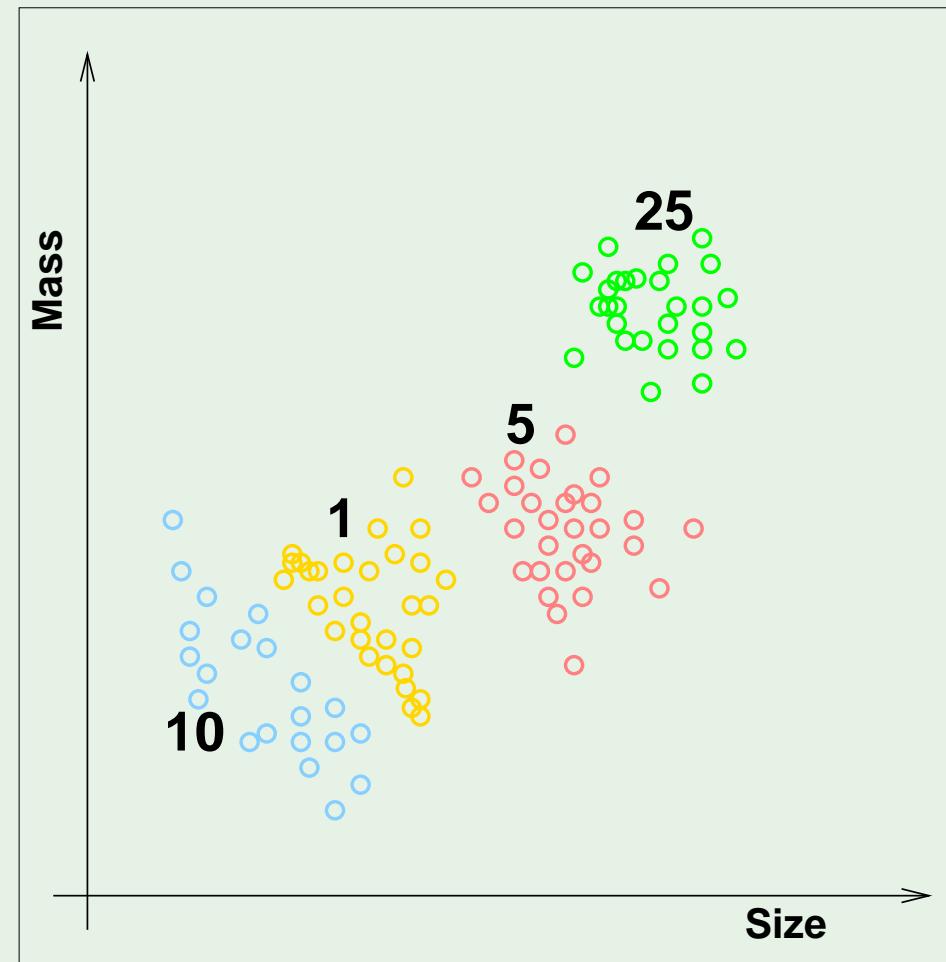
“using a set of observations to uncover an underlying process”

broad premise \implies many variations

- Supervised Learning
- Unsupervised Learning
- Reinforcement Learning

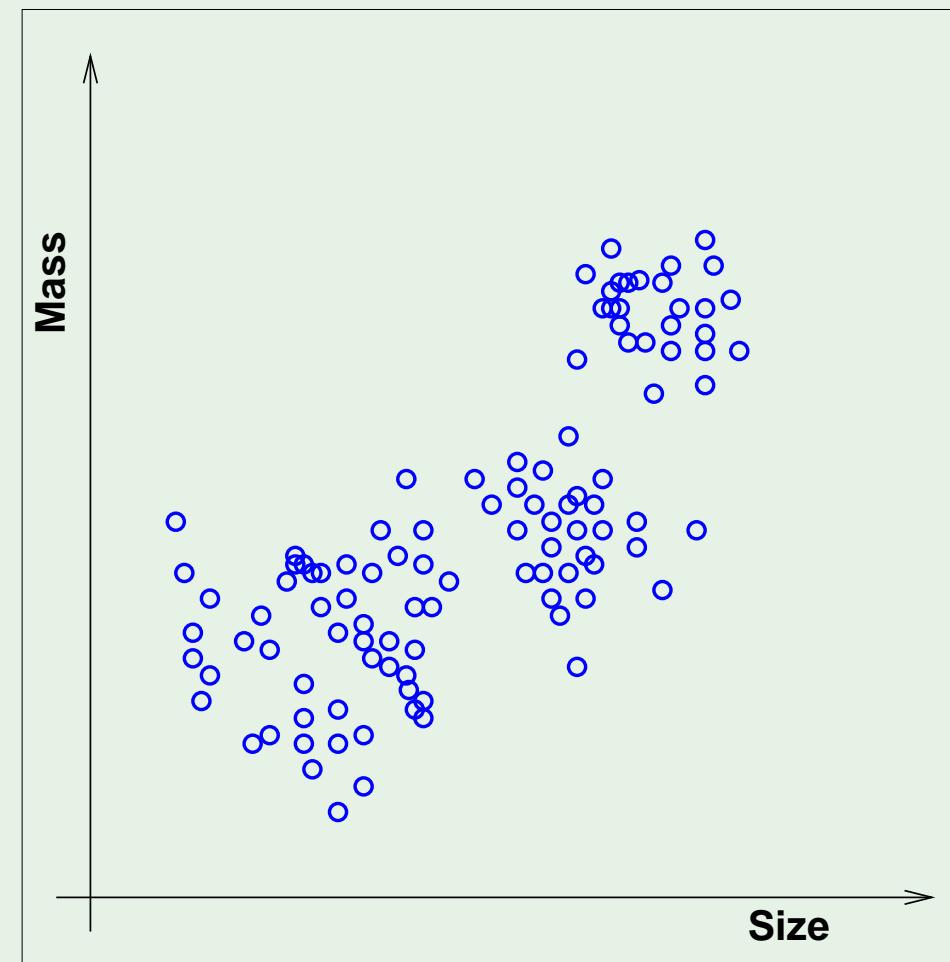
Supervised learning

Example from vending machines – coin recognition



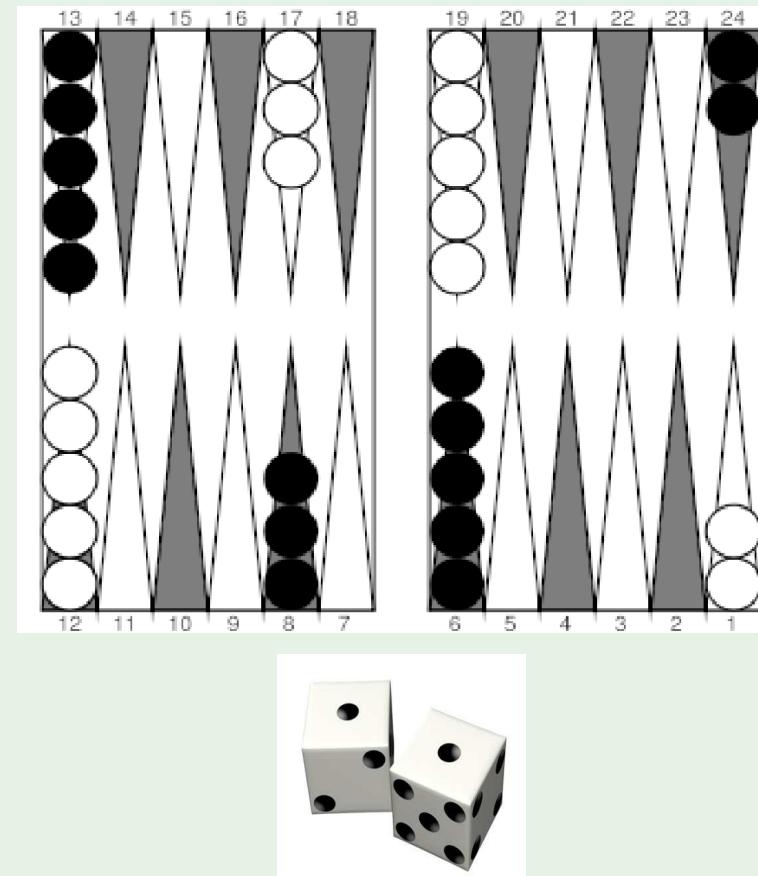
Unsupervised learning

Instead of **(input, correct output)**, we get **(input, ?)**



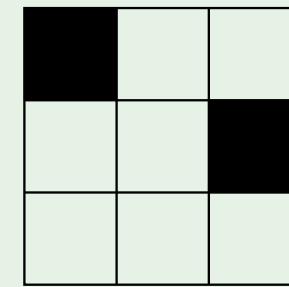
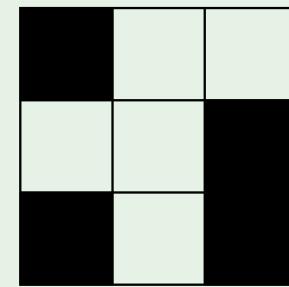
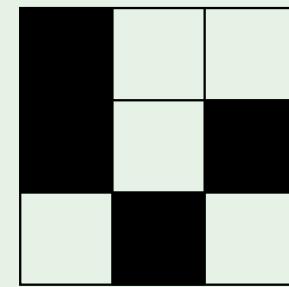
Reinforcement learning

Instead of **(input, correct output)**,
we get **(input, some output, grade for this output)**

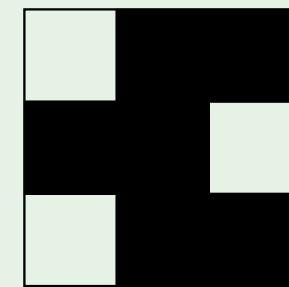
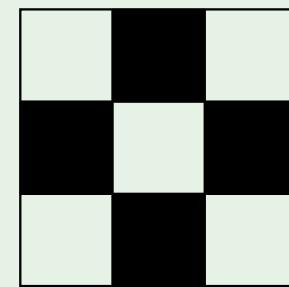
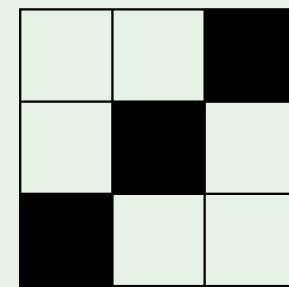


The world champion was
a neural network!

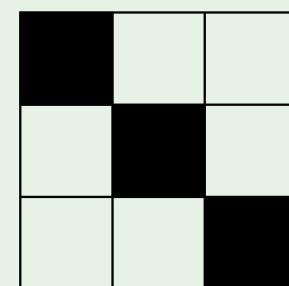
A Learning puzzle



$$f = -1$$



$$f = +1$$

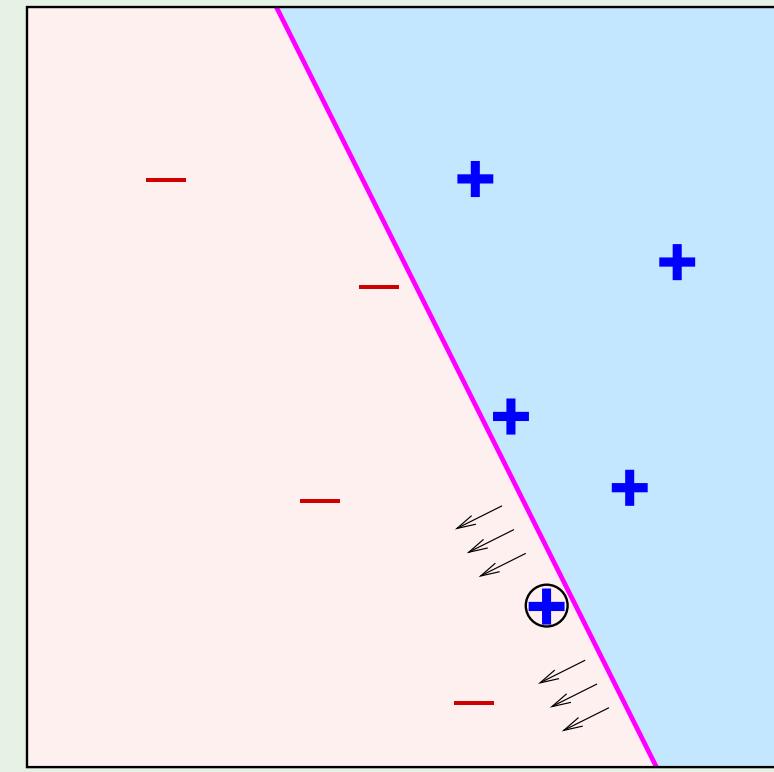


$$f = ?$$

Review of Lecture 1

- Learning is used when
 - A pattern exists
 - We cannot pin it down mathematically
 - We have data on it
- Focus on supervised learning
 - Unknown target function $y = f(\mathbf{x})$
 - Data set $(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)$
 - Learning algorithm picks $g \approx f$ from a hypothesis set \mathcal{H}

Example: Perceptron Learning Algorithm



- Learning an unknown function?
 - Impossible 😞. The function can assume any value outside the data we have.
 - So what now?

Learning From Data

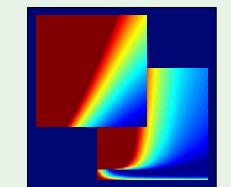
Yaser S. Abu-Mostafa
California Institute of Technology

Lecture 2: Is Learning Feasible?



Sponsored by Caltech's Provost Office, E&AS Division, and IST

• Thursday, April 5, 2012



Feasibility of learning - Outline

- Probability to the rescue
- Connection to learning
- Connection to *real* learning
- A dilemma and a solution

A related experiment

- Consider a 'bin' with **red** and **green** marbles.

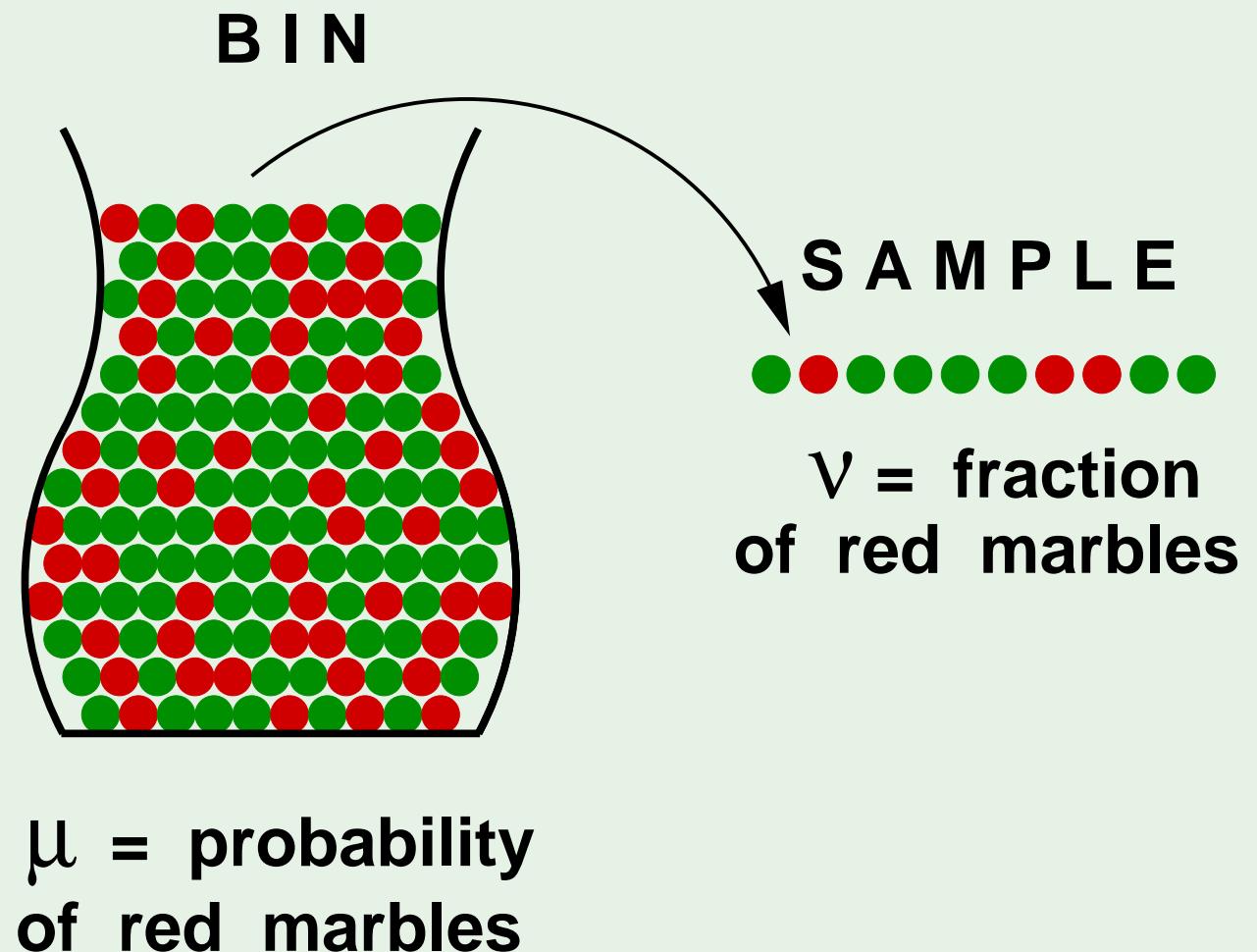
$$\mathbb{P}[\text{ picking a } \text{red} \text{ marble }] = \mu$$

$$\mathbb{P}[\text{ picking a } \text{green} \text{ marble}] = 1 - \mu$$

- The value of μ is unknown to us.

- We pick N marbles independently.

- The fraction of **red** marbles in sample = ν



Does ν say anything about μ ?

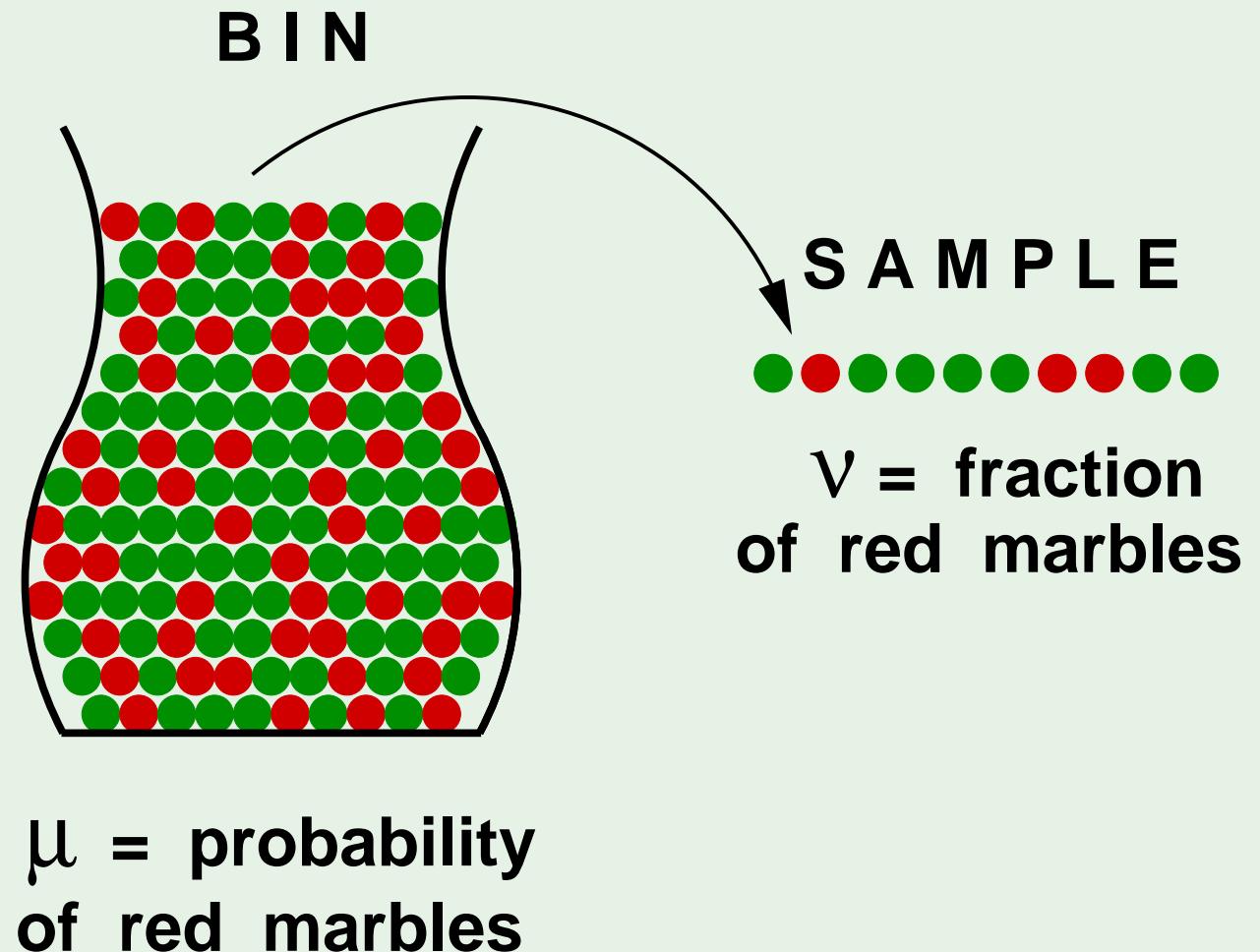
No!

Sample can be mostly green while bin is mostly red.

Yes!

Sample frequency ν is likely close to bin frequency μ .

possible versus probable



What does ν say about μ ?

In a big sample (large N), ν is probably close to μ (within ϵ).

Formally,

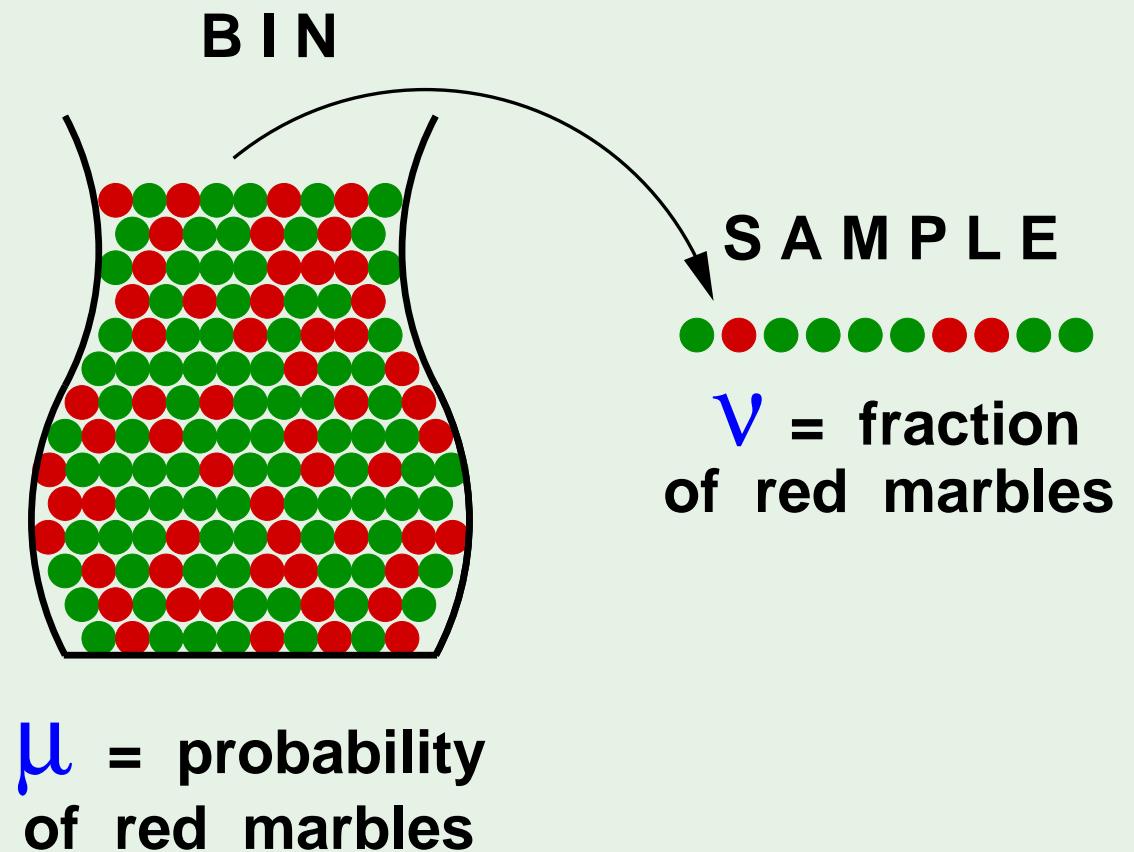
$$\mathbb{P} [|\nu - \mu| > \epsilon] \leq 2e^{-2\epsilon^2 N}$$

This is called **Hoeffding's Inequality**.

In other words, the statement " $\mu = \nu$ " is P.A.C.

$$\mathbb{P} [|\nu - \mu| > \epsilon] \leq 2e^{-2\epsilon^2 N}$$

- Valid for all N and ϵ
- Bound does not depend on μ
- Tradeoff: N , ϵ , and the bound.
- $\nu \approx \mu \implies \mu \approx \nu \quad \text{☺}$



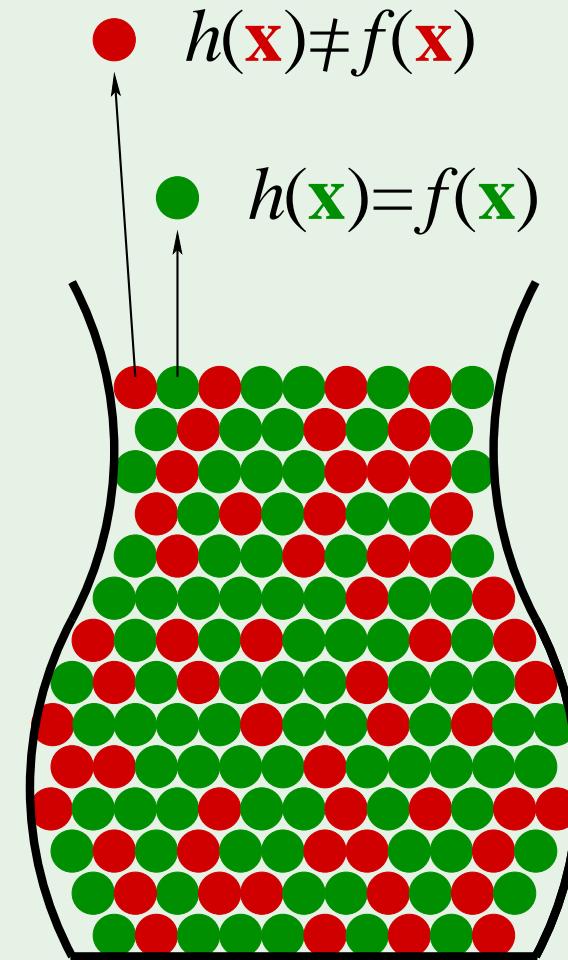
Connection to learning

Bin: The unknown is a number μ

Learning: The unknown is a function $f : \mathcal{X} \rightarrow \mathcal{Y}$

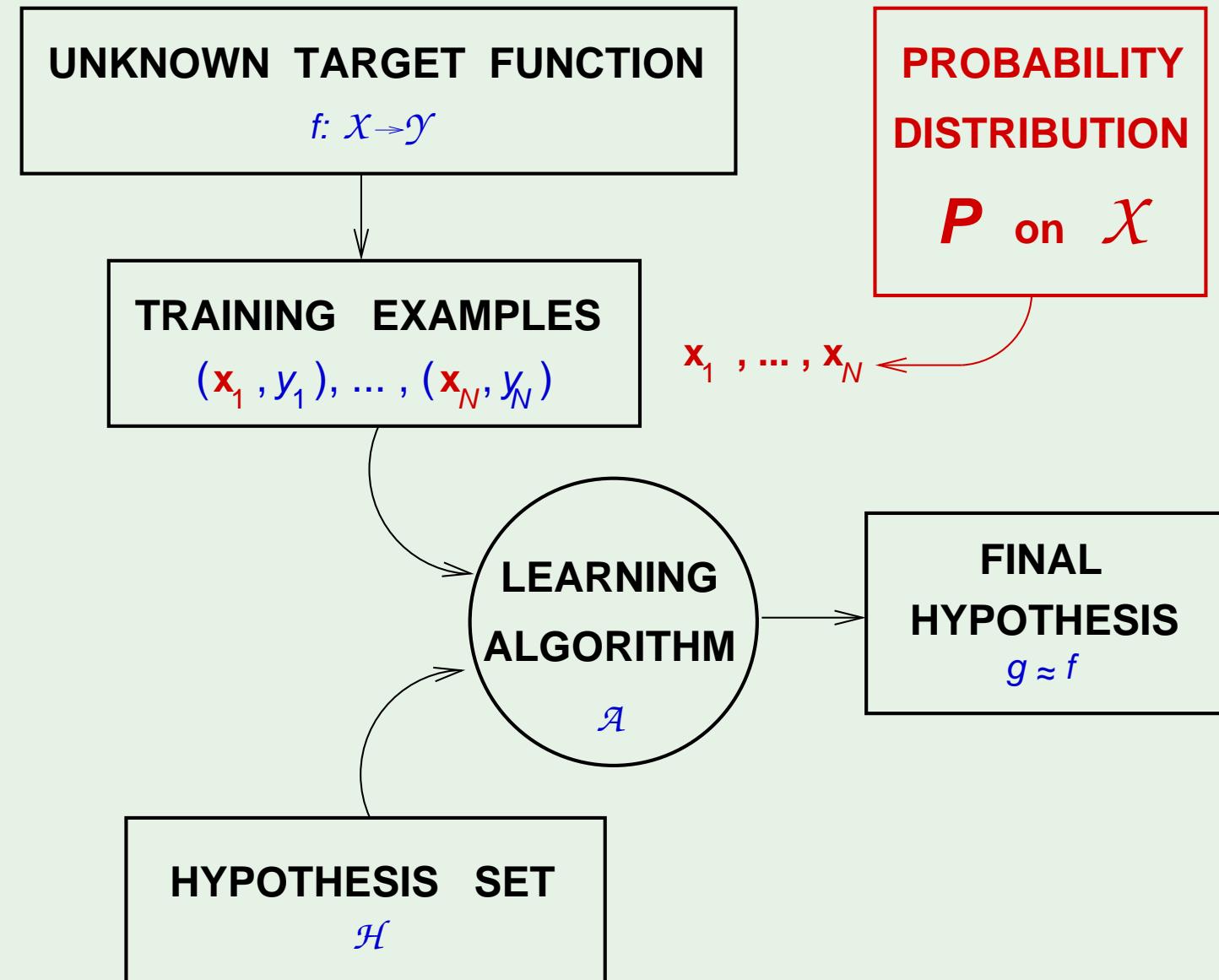
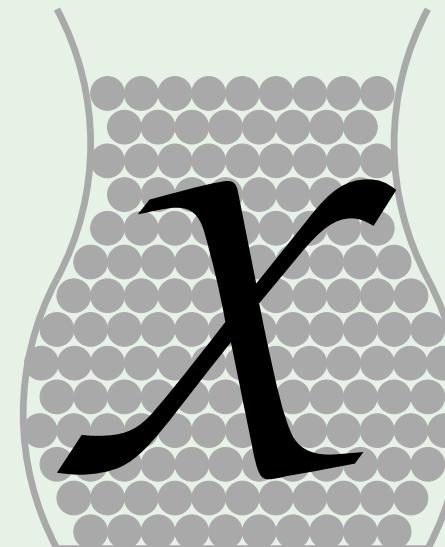
Each marble \bullet is a point $\mathbf{x} \in \mathcal{X}$

- : Hypothesis got it **right** $h(\mathbf{x})=f(\mathbf{x})$
- : Hypothesis got it **wrong** $h(\mathbf{x})\neq f(\mathbf{x})$



Back to the learning diagram

The bin analogy:



Are we done?

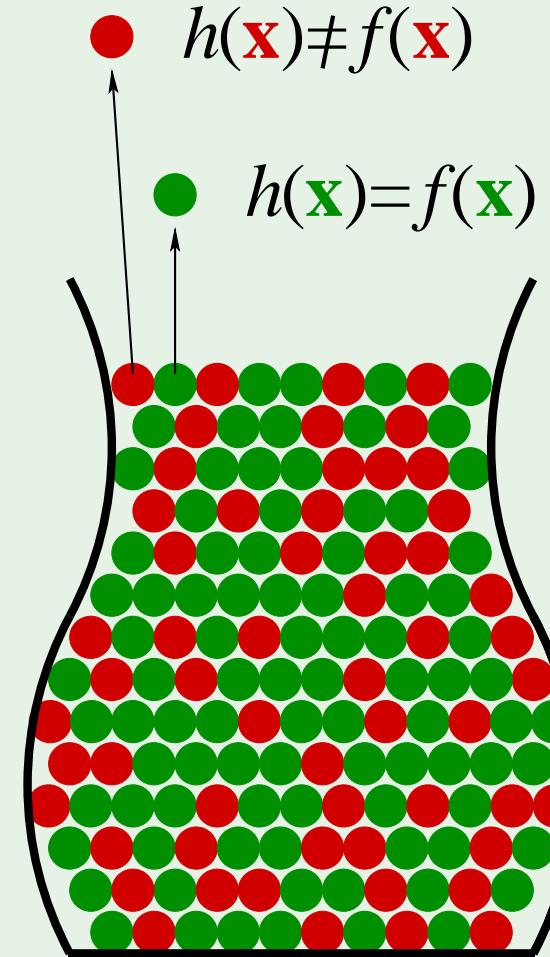
Not so fast! h is fixed.

For this h , ν generalizes to μ .

'verification' of h , not **learning**

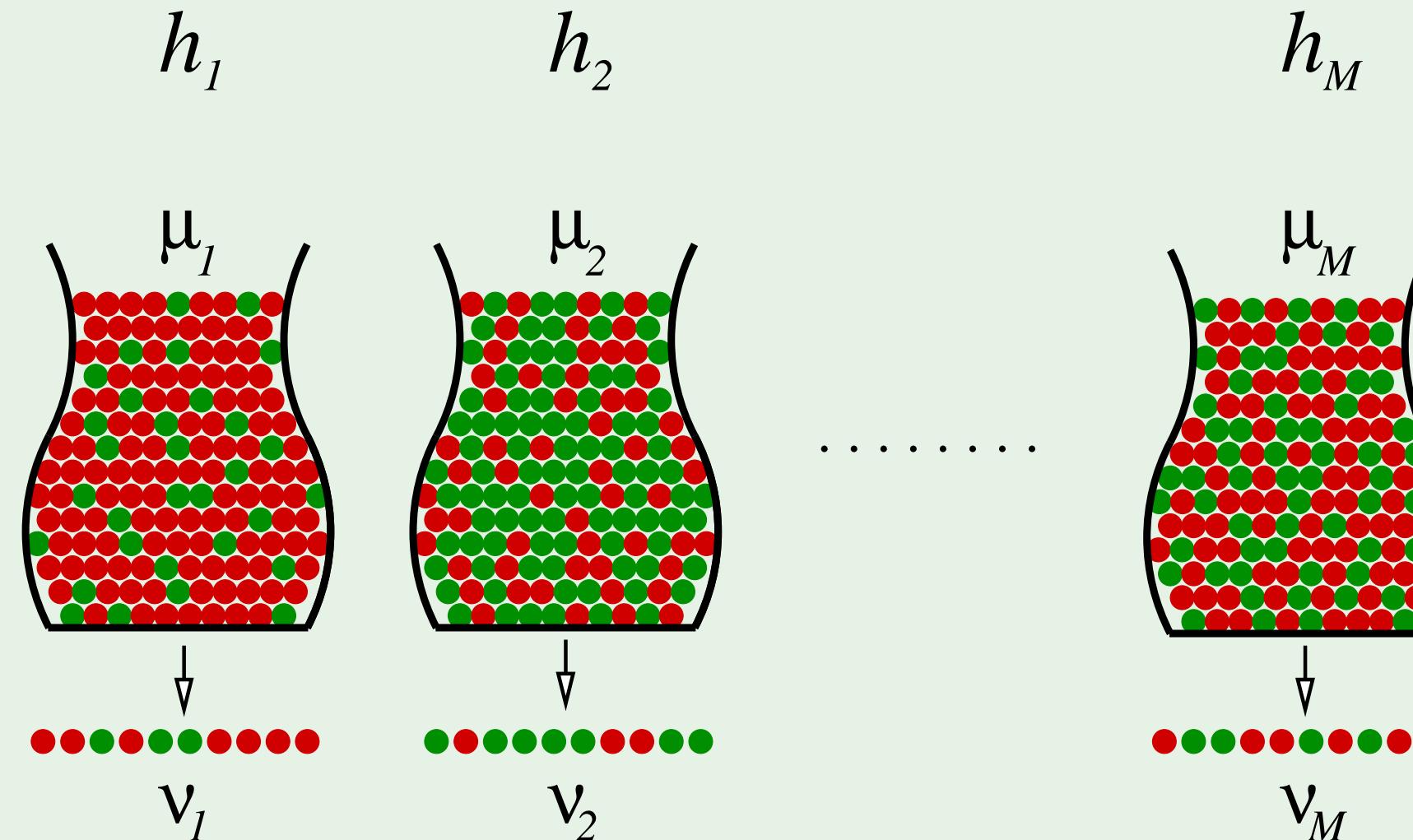
No guarantee ν will be small.

We need to **choose** from multiple h 's.



Multiple bins

Generalizing the bin model to more than one hypothesis:



Notation for learning

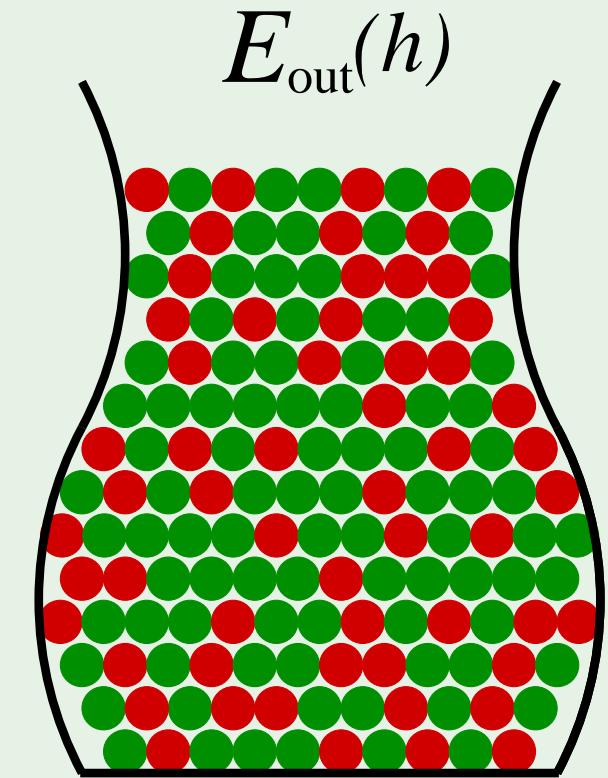
Both μ and ν depend on which hypothesis h

ν is '**in sample**' denoted by $E_{\text{in}}(h)$

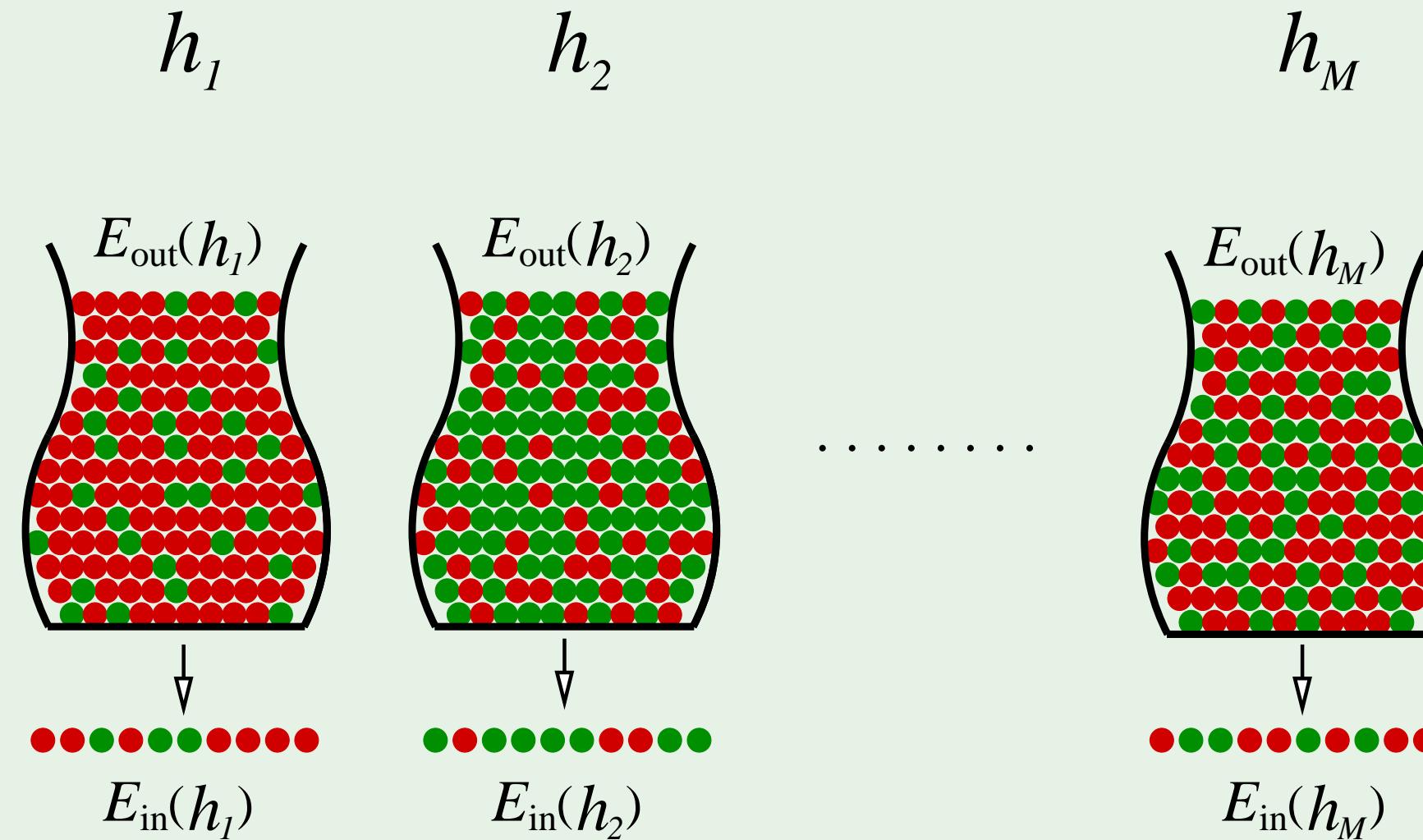
μ is '**out of sample**' denoted by $E_{\text{out}}(h)$

The Hoeffding inequality becomes:

$$\mathbb{P} [|E_{\text{in}}(h) - E_{\text{out}}(h)| > \epsilon] \leq 2e^{-2\epsilon^2 N}$$



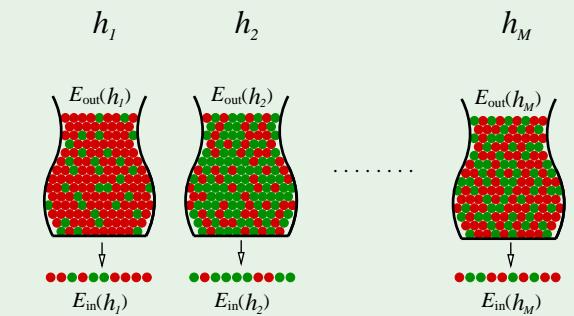
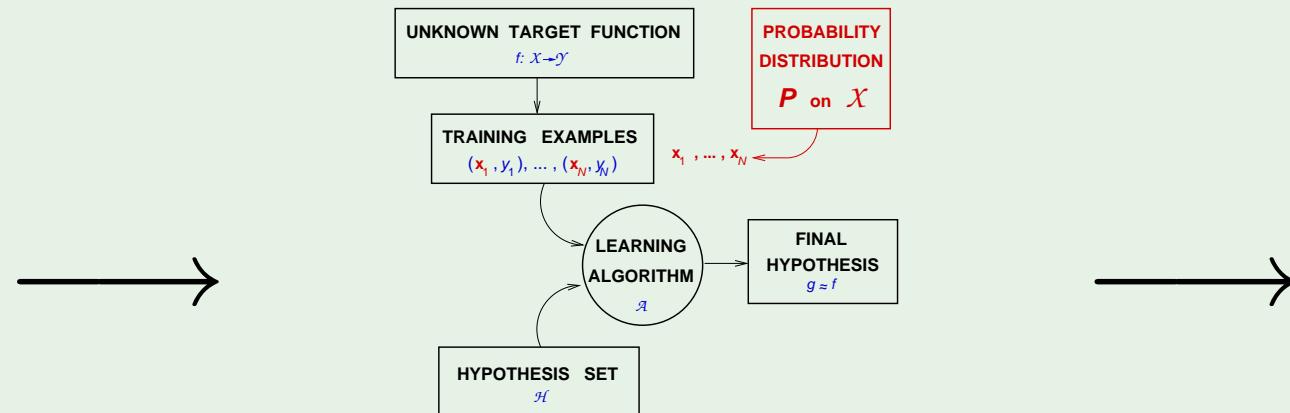
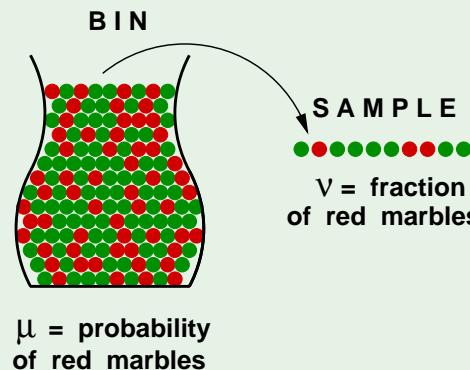
Notation with multiple bins



Are we done already? 😊

Not so fast!! Hoeffding doesn't apply to multiple bins.

What?



Coin analogy

Question: If you toss a fair coin 10 times, what is the probability that you will get 10 heads?

Answer: $\approx 0.1\%$

Question: If you toss 1000 fair coins 10 times each, what is the probability that some coin will get 10 heads?

Answer: $\approx 63\%$

From coins to learning

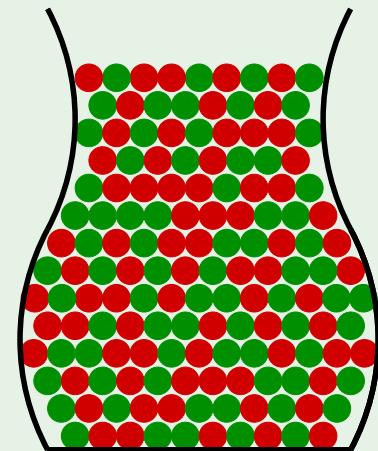
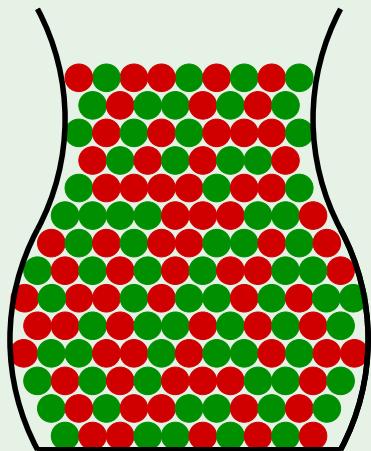
hi



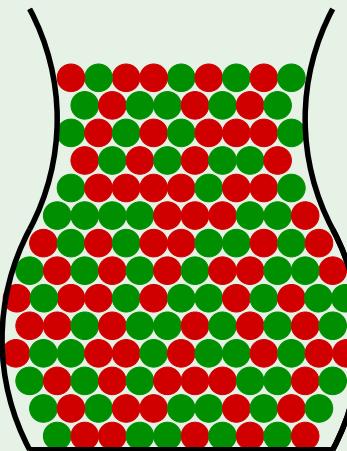
• • • • •



• • • • • • • • •



• • • • •



• • • • • • • • •

• • • • • • •

• • • • • • •



B I N G O ?

Hi

A simple solution

$$\begin{aligned}\mathbb{P}[|E_{\text{in}}(g) - E_{\text{out}}(g)| > \epsilon] &\leq \mathbb{P}[|E_{\text{in}}(h_1) - E_{\text{out}}(h_1)| > \epsilon \\ &\quad \text{or } |E_{\text{in}}(h_2) - E_{\text{out}}(h_2)| > \epsilon \\ &\quad \dots \\ &\quad \text{or } |E_{\text{in}}(h_M) - E_{\text{out}}(h_M)| > \epsilon] \\ &\leq \sum_{m=1}^M \mathbb{P}[|E_{\text{in}}(h_m) - E_{\text{out}}(h_m)| > \epsilon]\end{aligned}$$

The final verdict

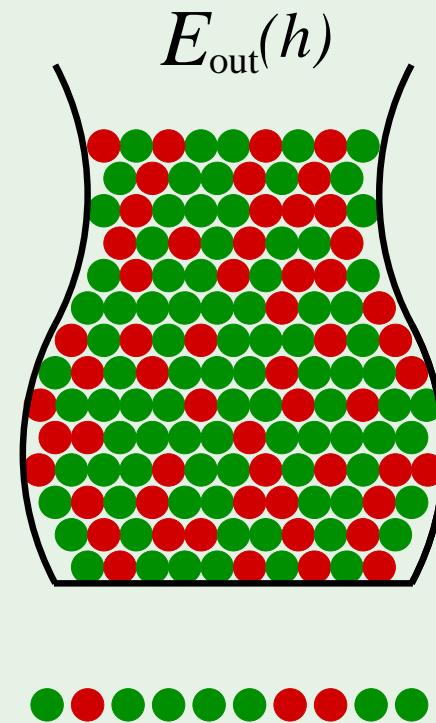
$$\begin{aligned}\mathbb{P}[|E_{\text{in}}(g) - E_{\text{out}}(g)| > \epsilon] &\leq \sum_{m=1}^M \mathbb{P}[|E_{\text{in}}(h_m) - E_{\text{out}}(h_m)| > \epsilon] \\ &\leq \sum_{m=1}^M 2e^{-2\epsilon^2 N}\end{aligned}$$

$$\mathbb{P}[|E_{\text{in}}(g) - E_{\text{out}}(g)| > \epsilon] \leq 2M e^{-2\epsilon^2 N}$$

Review of Lecture 2

Is Learning feasible?

Yes, in a **probabilistic** sense.



$$\mathbb{P} [|E_{\text{in}}(h) - E_{\text{out}}(h)| > \epsilon] \leq 2e^{-2\epsilon^2 N}$$

Since g has to be one of h_1, h_2, \dots, h_M , we conclude that

If:

$$|E_{\text{in}}(g) - E_{\text{out}}(g)| > \epsilon$$

Then:

$$|E_{\text{in}}(h_1) - E_{\text{out}}(h_1)| > \epsilon \quad \text{or}$$

$$|E_{\text{in}}(h_2) - E_{\text{out}}(h_2)| > \epsilon \quad \text{or}$$

...

$$|E_{\text{in}}(h_M) - E_{\text{out}}(h_M)| > \epsilon$$

This gives us an added **M** factor.

Learning From Data

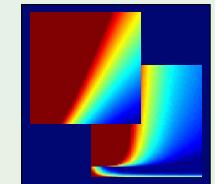
Yaser S. Abu-Mostafa
California Institute of Technology

Lecture 3: Linear Models I



Sponsored by Caltech's Provost Office, E&AS Division, and IST

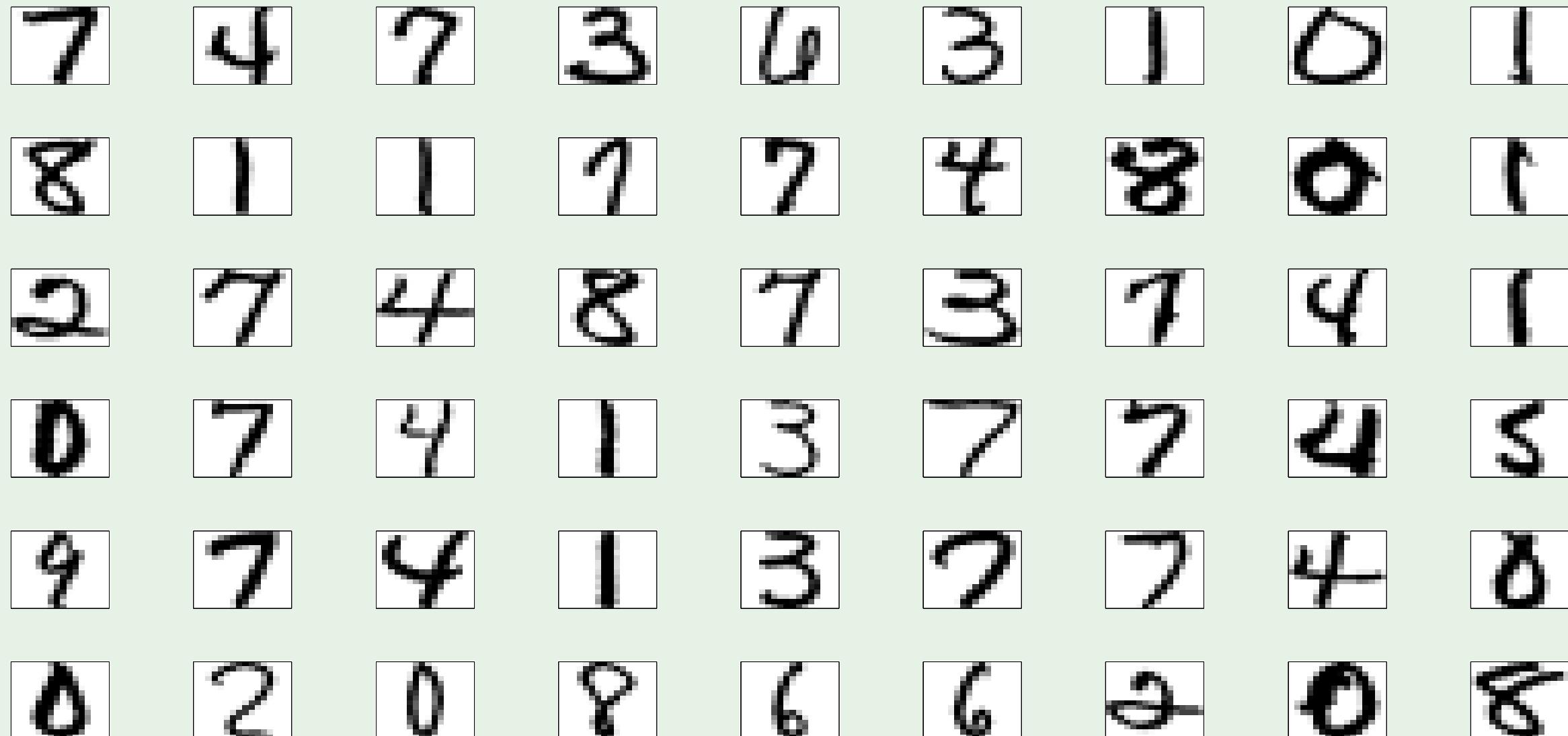
• Tuesday, April 10, 2012



Outline

- Input representation
- Linear Classification
- Linear Regression
- Nonlinear Transformation

A real data set



Input representation

'raw' input $\mathbf{x} = (x_0, x_1, x_2, \dots, x_{256})$

linear model: $(w_0, w_1, w_2, \dots, w_{256})$

Features: Extract useful information, e.g.,

intensity and symmetry $\mathbf{x} = (x_0, x_1, x_2)$

linear model: (w_0, w_1, w_2)

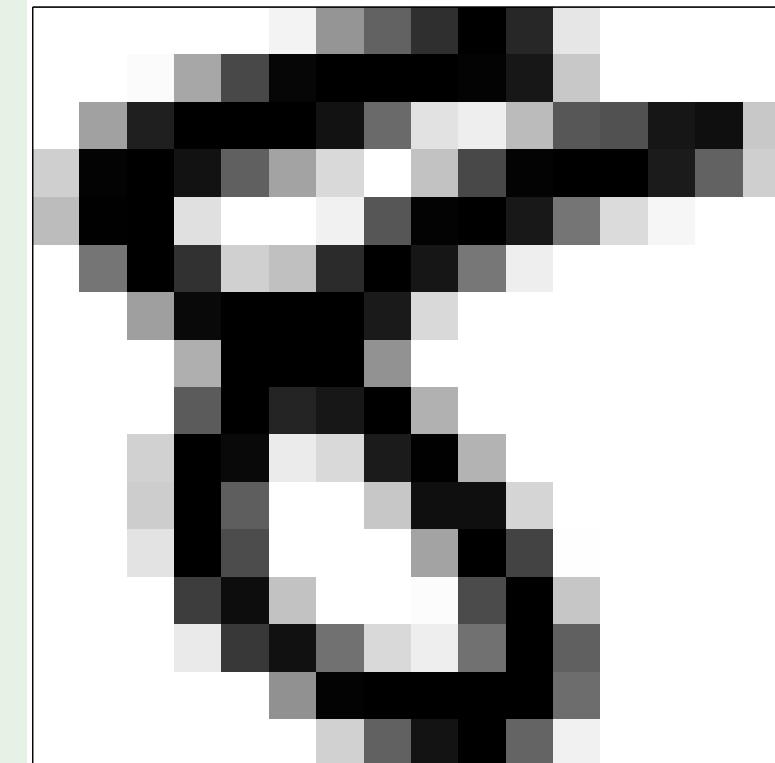
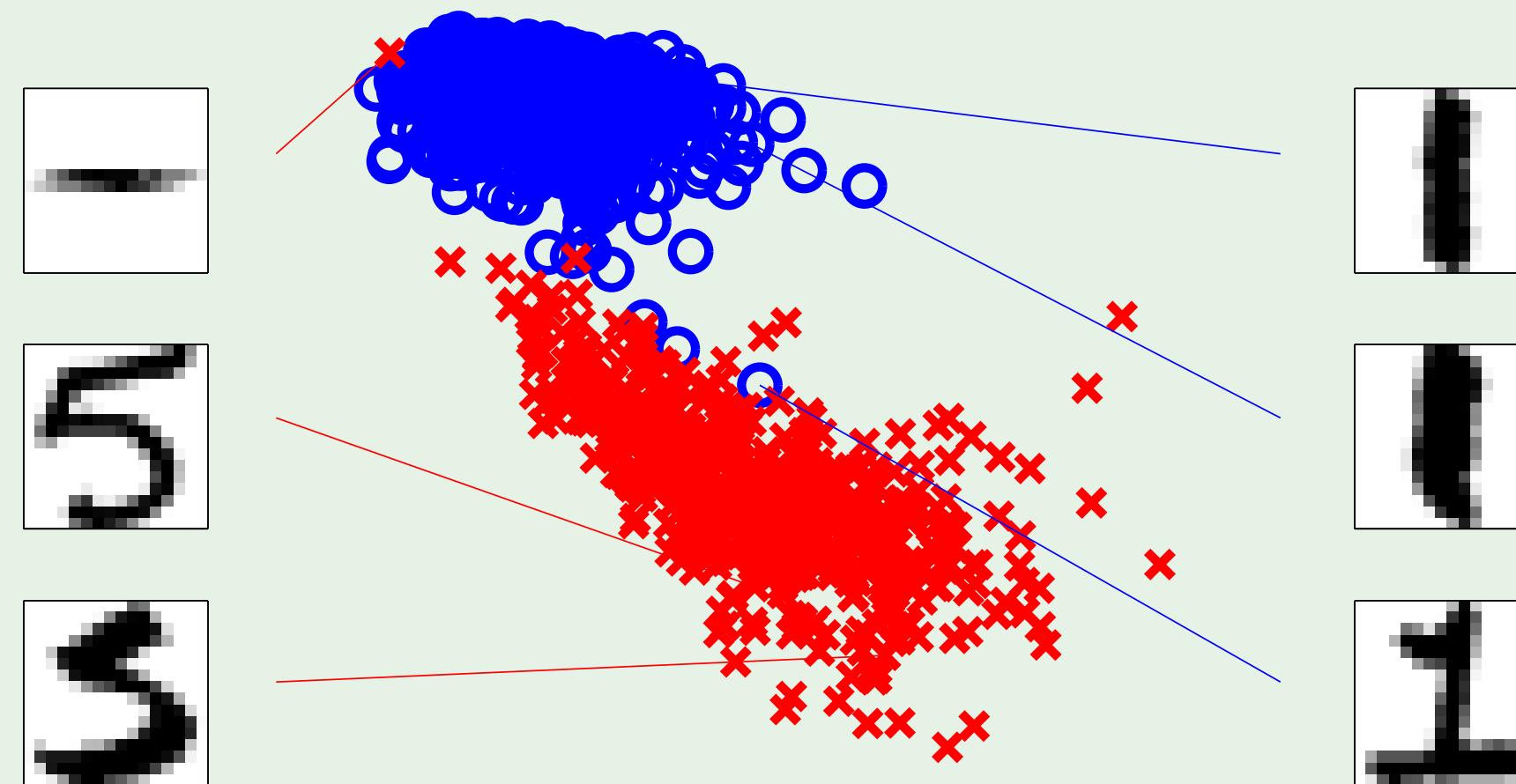


Illustration of features

$$\mathbf{x} = (x_0, x_1, x_2)$$

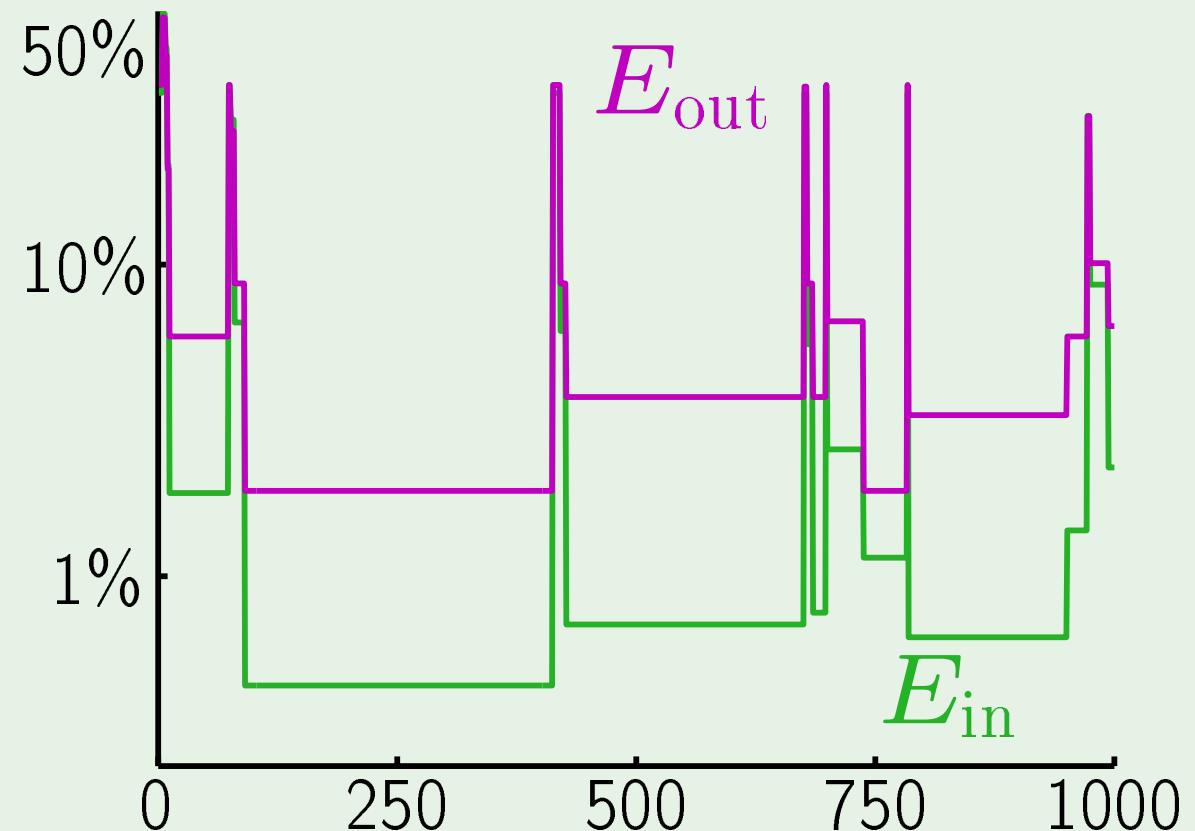
x_1 : intensity

x_2 : symmetry

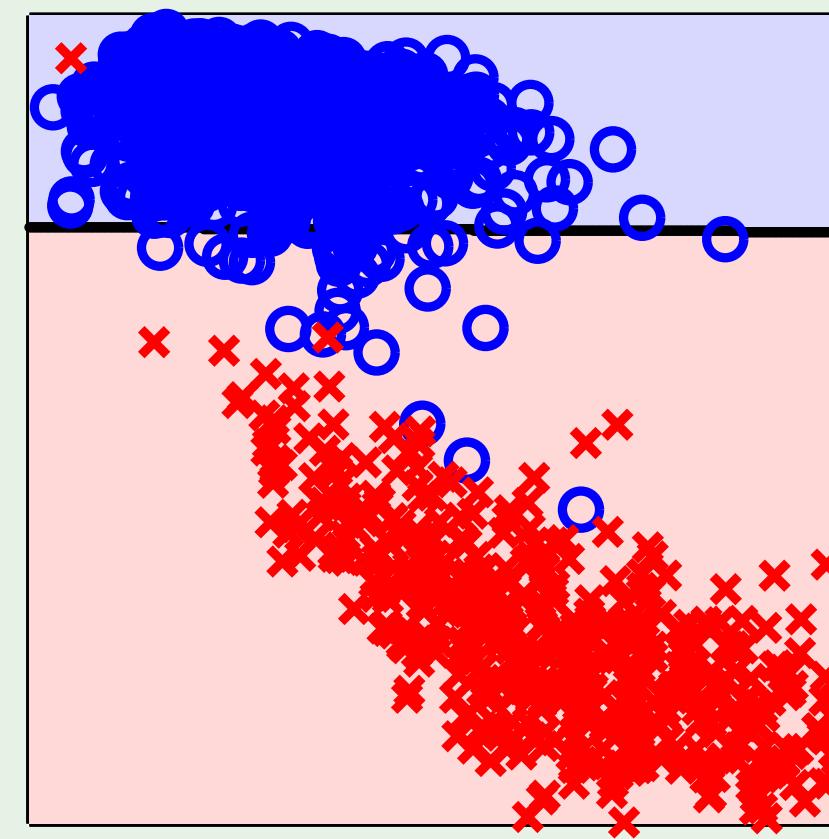


What PLA does

Evolution of E_{in} and E_{out}

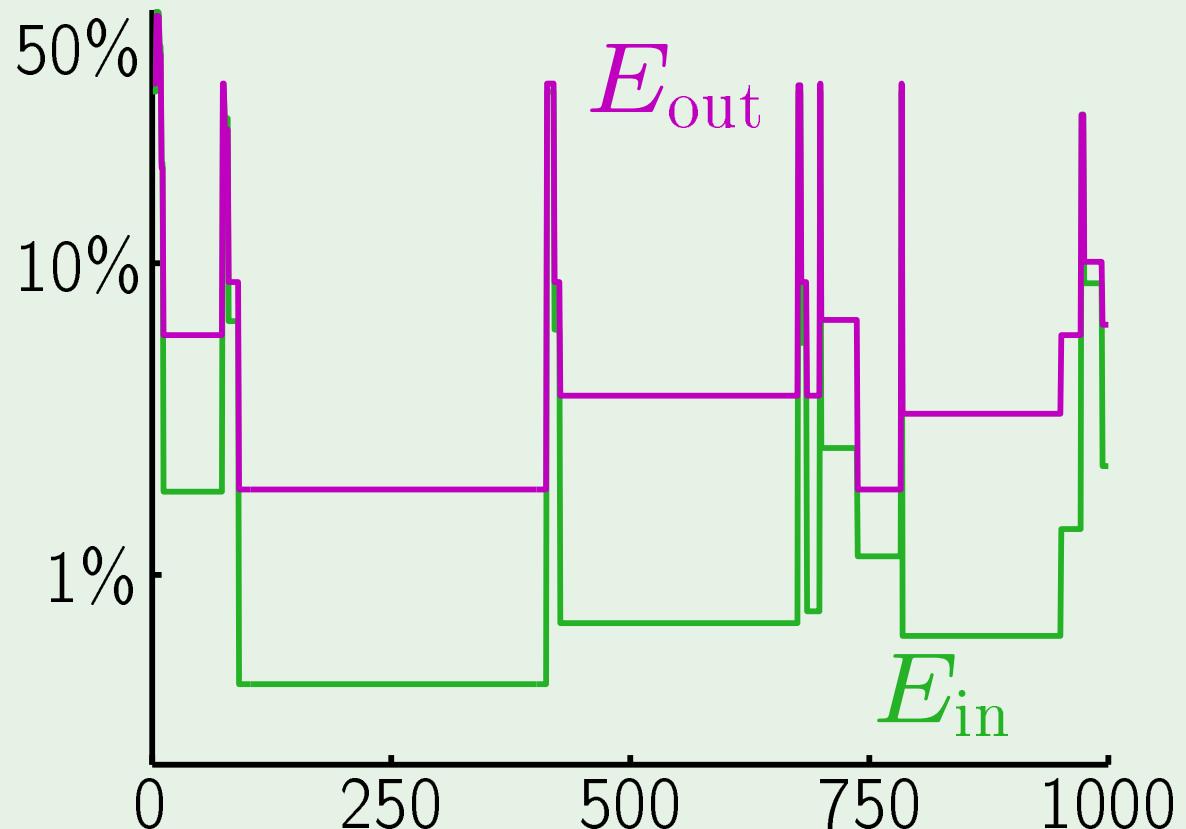


Final perceptron boundary

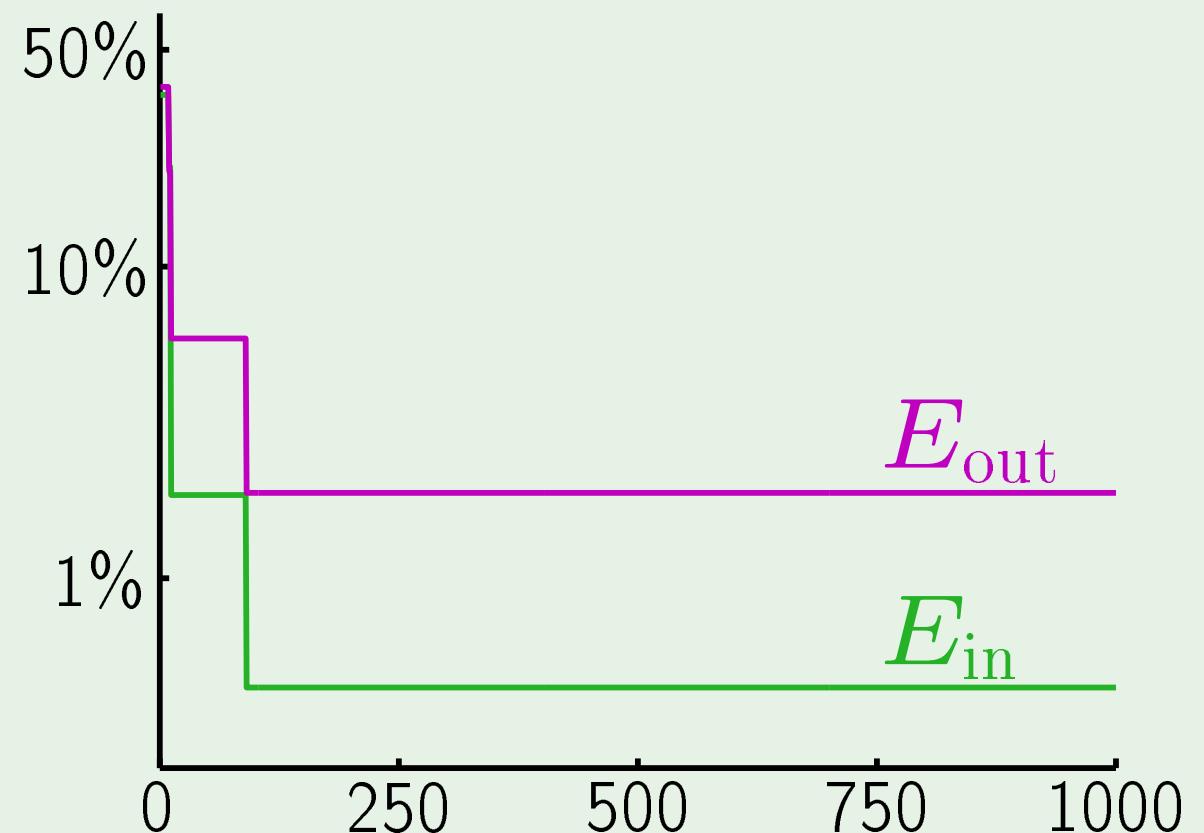


The ‘pocket’ algorithm

PLA:

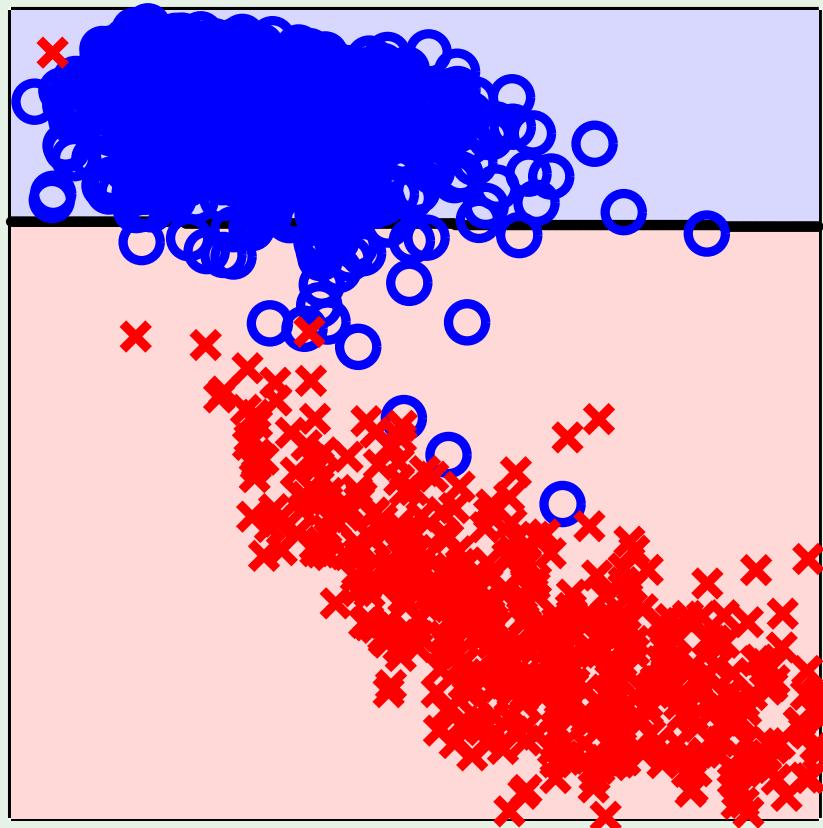


Pocket:

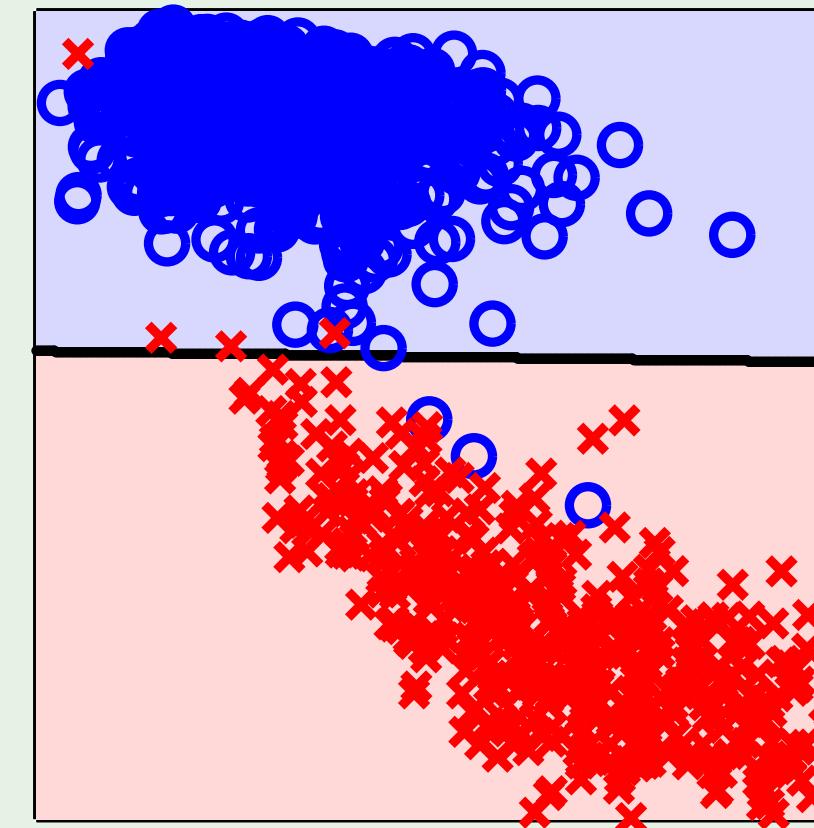


Classification boundary - PLA versus Pocket

PLA:



Pocket:



Outline

- Input representation
- Linear Classification
- Linear Regression **regression \equiv real-valued output**
- Nonlinear Transformation

Credit again

Classification: Credit approval (yes/no)

Regression: Credit line (dollar amount)

Input: $\mathbf{x} =$

age	23 years
annual salary	\$30,000
years in residence	1 year
years in job	1 year
current debt	\$15,000
...	...

Linear regression output: $h(\mathbf{x}) = \sum_{i=0}^d w_i x_i = \mathbf{w}^\top \mathbf{x}$

The data set

Credit officers decide on credit lines:

$$(\mathbf{x}_1, \textcolor{blue}{y}_1), (\mathbf{x}_2, \textcolor{blue}{y}_2), \dots, (\mathbf{x}_N, \textcolor{blue}{y}_N)$$

$\textcolor{blue}{y}_n \in \mathbb{R}$ is the credit line for customer \mathbf{x}_n .

Linear regression tries to replicate that.

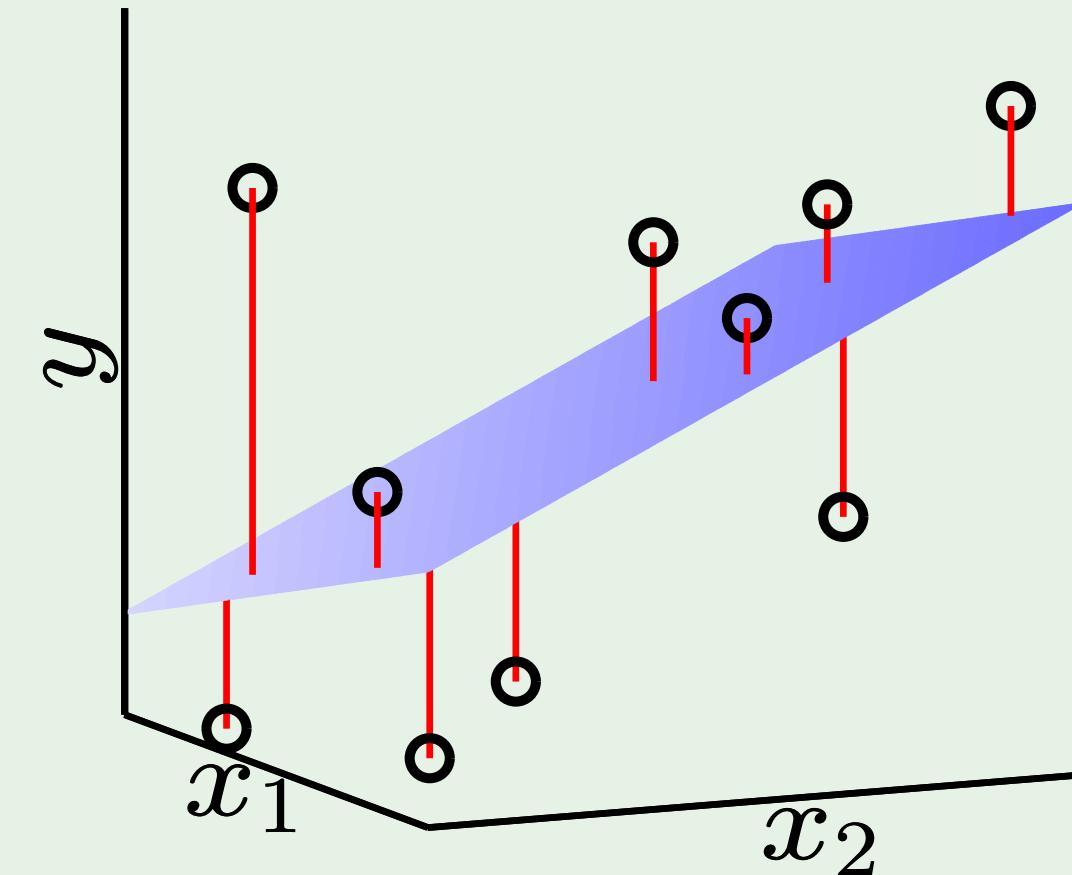
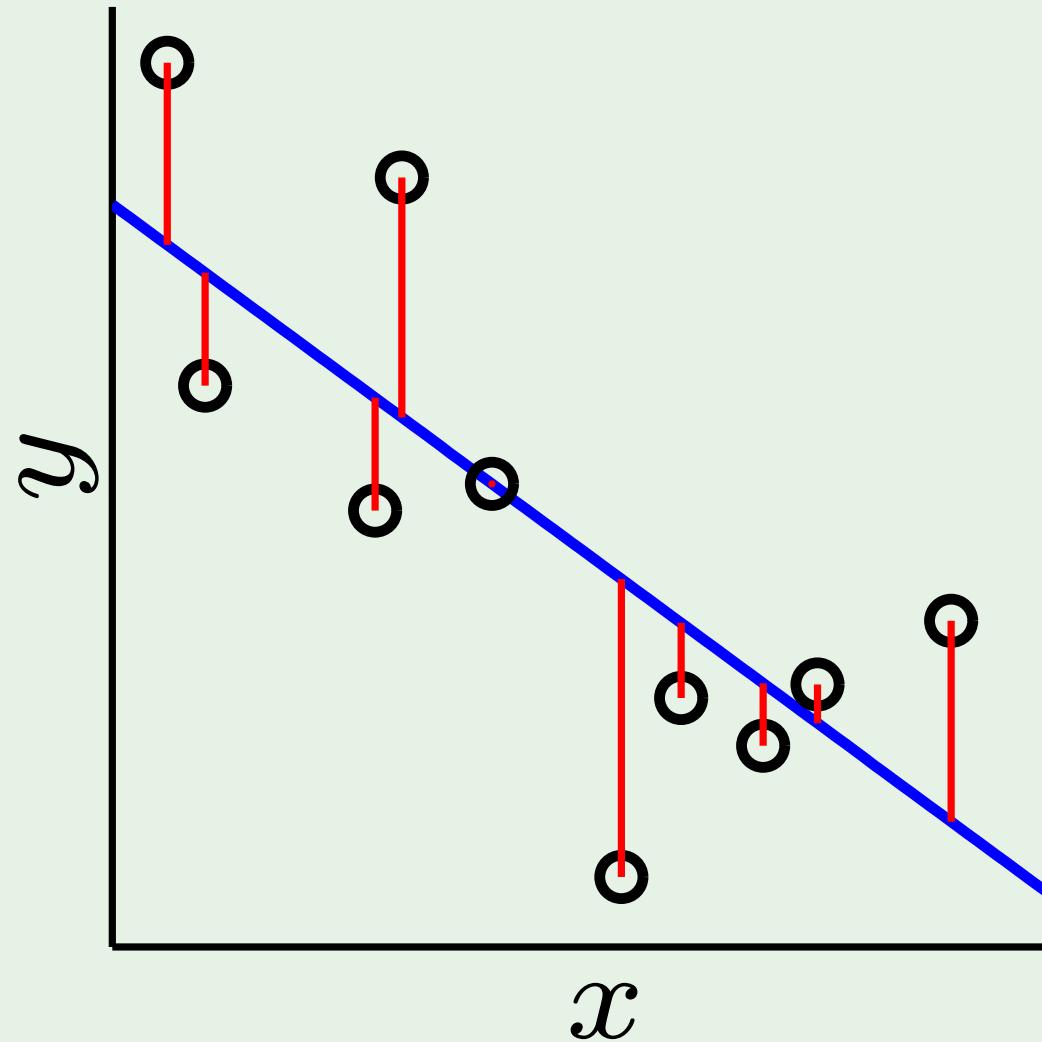
How to measure the error

How well does $h(\mathbf{x}) = \mathbf{w}^\top \mathbf{x}$ approximate $f(\mathbf{x})$?

In linear regression, we use squared error $(h(\mathbf{x}) - f(\mathbf{x}))^2$

in-sample error: $E_{\text{in}}(h) = \frac{1}{N} \sum_{n=1}^N (h(\mathbf{x}_n) - y_n)^2$

Illustration of linear regression



The expression for E_{in}

$$\begin{aligned} E_{\text{in}}(\mathbf{w}) &= \frac{1}{N} \sum_{n=1}^N (\mathbf{w}^\top \mathbf{x}_n - y_n)^2 \\ &= \frac{1}{N} \|\mathbf{X}\mathbf{w} - \mathbf{y}\|^2 \end{aligned}$$

where

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1^\top \\ \mathbf{x}_2^\top \\ \vdots \\ \mathbf{x}_N^\top \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}$$

Minimizing E_{in}

$$E_{\text{in}}(\mathbf{w}) = \frac{1}{N} \|\mathbf{X}\mathbf{w} - \mathbf{y}\|^2$$

$$\nabla E_{\text{in}}(\mathbf{w}) = \frac{2}{N} \mathbf{X}^\top (\mathbf{X}\mathbf{w} - \mathbf{y}) = \mathbf{0}$$

$$\mathbf{X}^\top \mathbf{X}\mathbf{w} = \mathbf{X}^\top \mathbf{y}$$

$$\mathbf{w} = \mathbf{X}^\dagger \mathbf{y} \quad \text{where} \quad \mathbf{X}^\dagger = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top$$

\mathbf{X}^\dagger is the '**pseudo-inverse**' of \mathbf{X}

The pseudo-inverse

$$\mathbf{X}^\dagger = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top$$

$$\left(\begin{bmatrix} & \\ & \end{bmatrix}_{d+1 \times d+1} \right)^{-1} \begin{bmatrix} & \\ & \end{bmatrix}_{d+1 \times N}$$

$d+1 \times N$

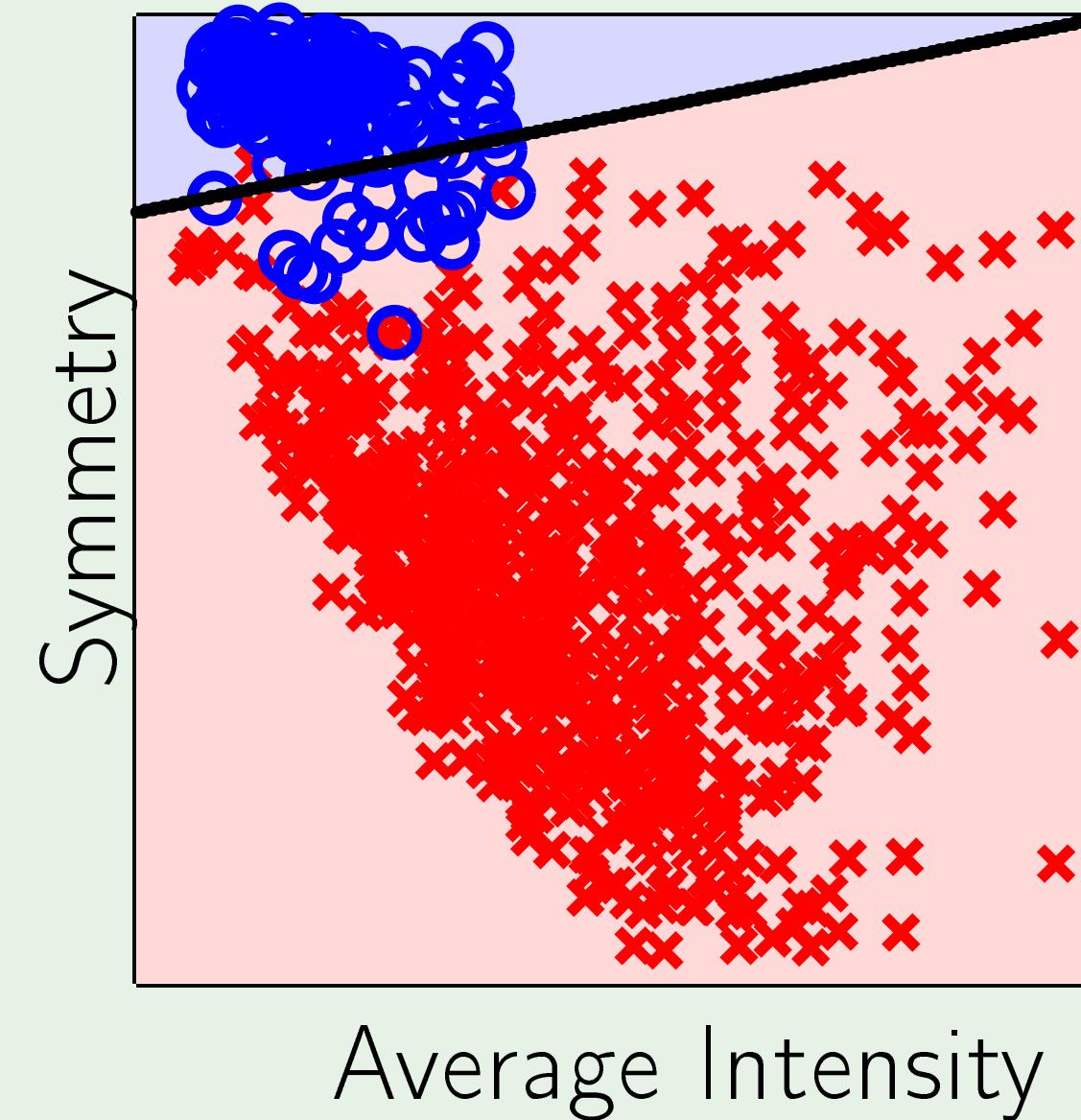
The linear regression algorithm

- 1: Construct the matrix \mathbf{X} and the vector \mathbf{y} from the data set $(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)$ as follows

$$\mathbf{X} = \underbrace{\begin{bmatrix} \mathbf{x}_1^\top \\ \mathbf{x}_2^\top \\ \vdots \\ \mathbf{x}_N^\top \end{bmatrix}}_{\text{input data matrix}}, \quad \mathbf{y} = \underbrace{\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}}_{\text{target vector}}.$$

- 2: Compute the pseudo-inverse $\mathbf{X}^\dagger = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top$.
- 3: Return $\mathbf{w} = \mathbf{X}^\dagger \mathbf{y}$.

Linear regression boundary

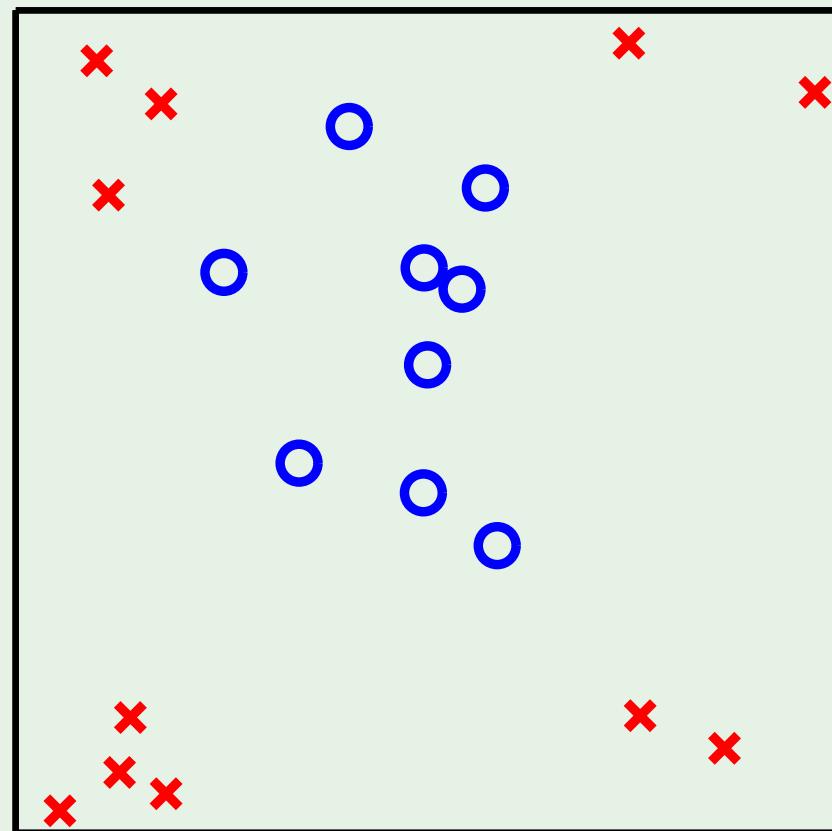


Outline

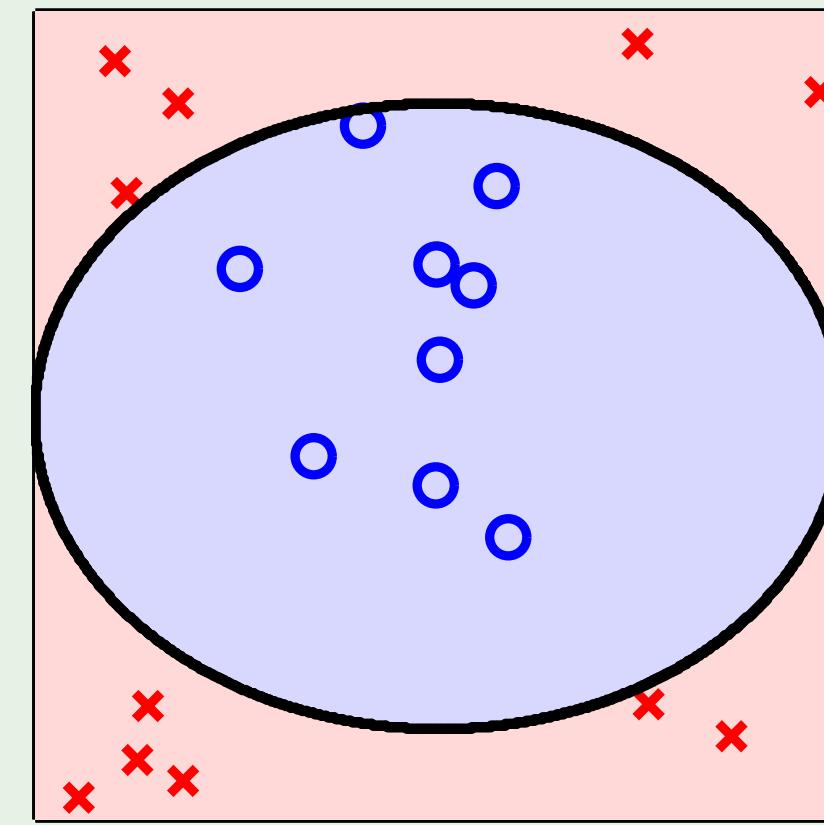
- Input representation
- Linear Classification
- Linear Regression
- Nonlinear Transformation

Linear is limited

Data:



Hypothesis:



Another example

Credit line is affected by ‘years in residence’

but **not** in a linear way!

Nonlinear $\llbracket x_i < 1 \rrbracket$ and $\llbracket x_i > 5 \rrbracket$ are better.

Can we do that with linear models?

Linear in what?

Linear regression implements

$$\sum_{i=0}^d \textcolor{red}{w}_i x_i$$

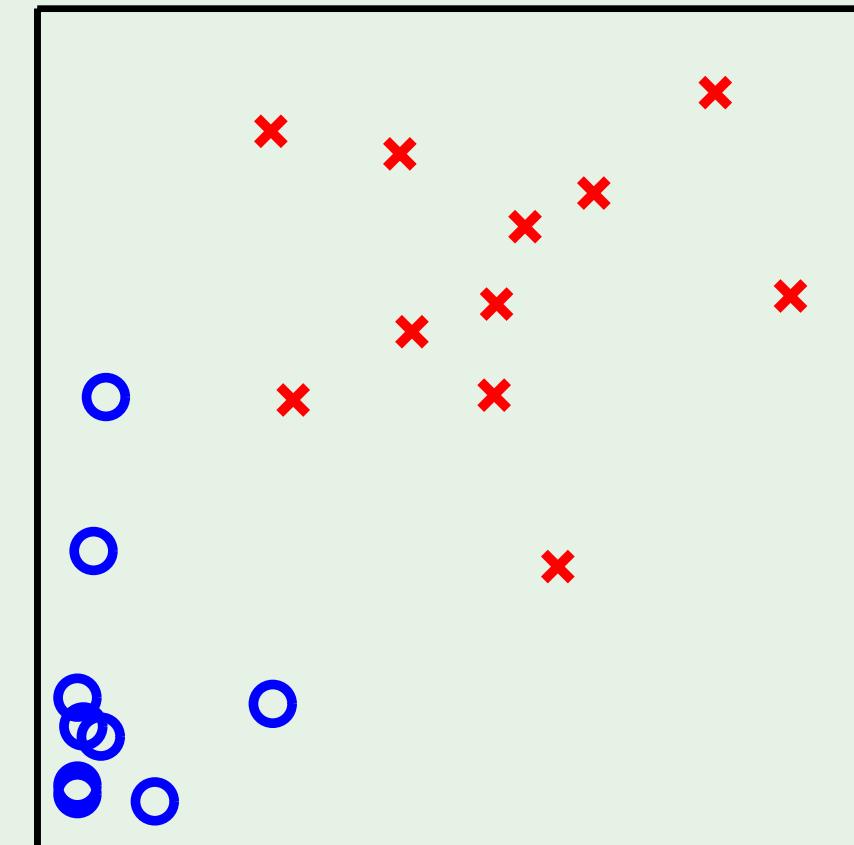
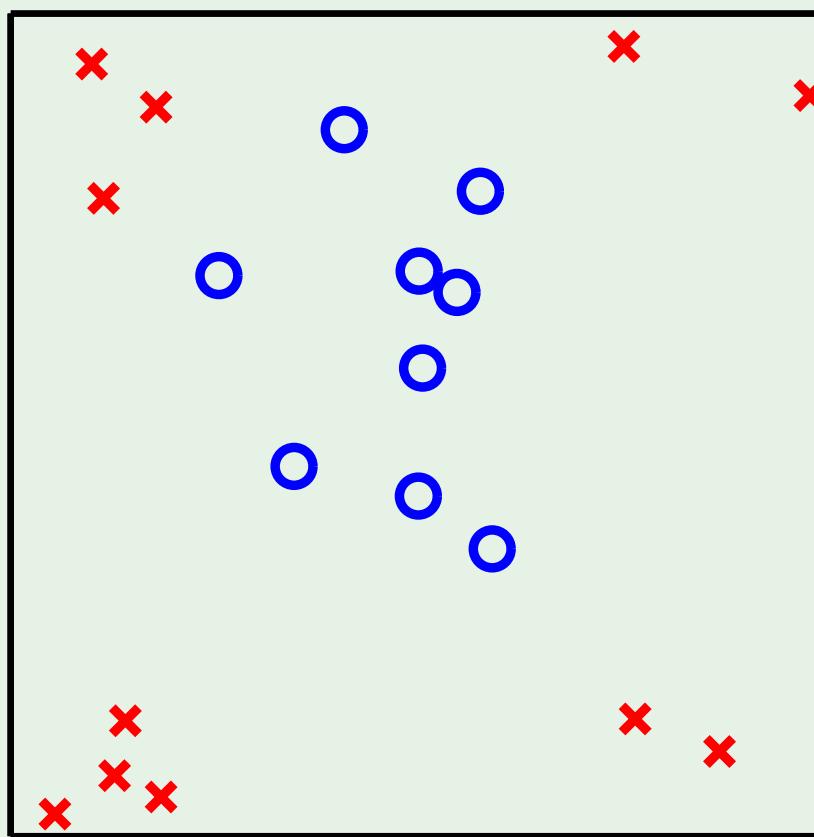
Linear classification implements

$$\text{sign} \left(\sum_{i=0}^d \textcolor{red}{w}_i x_i \right)$$

Algorithms work because of **linearity in the weights**

Transform the data nonlinearly

$$(x_1, x_2) \xrightarrow{\Phi} (x_1^2, x_2^2)$$



Review of Lecture 3

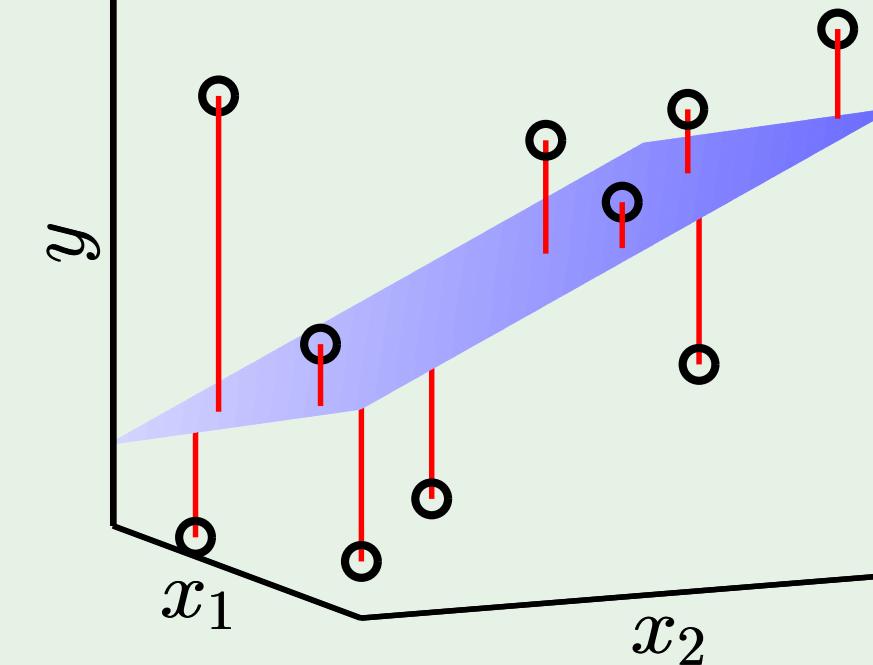
- Linear models use the ‘signal’: $\sum_{i=0}^d w_i x_i = \mathbf{w}^\top \mathbf{x}$

- Classification: $h(\mathbf{x}) = \text{sign}(\mathbf{w}^\top \mathbf{x})$
- Regression: $h(\mathbf{x}) = \mathbf{w}^\top \mathbf{x}$

- Linear regression algorithm:

$$\mathbf{w} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}$$

“one-step learning”



- Nonlinear transformation:
 - $\mathbf{w}^\top \mathbf{x}$ is linear in \mathbf{w}
 - Any $\mathbf{x} \xrightarrow{\Phi} \mathbf{z}$ preserves this linearity.
 - Example: $(x_1, x_2) \xrightarrow{\Phi} (x_1^2, x_2^2)$

Learning From Data

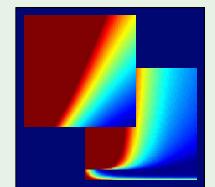
Yaser S. Abu-Mostafa
California Institute of Technology

Lecture 4: Error and Noise



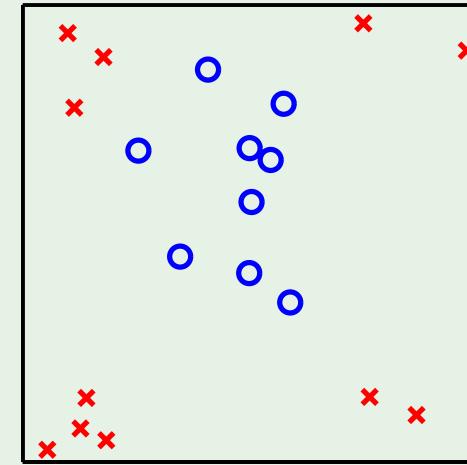
Sponsored by Caltech's Provost Office, E&AS Division, and IST

• Thursday, April 12, 2012



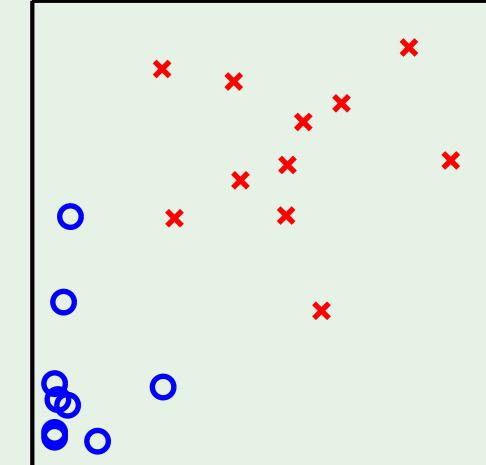
Outline

- Nonlinear transformation (continued)
- Error measures
- Noisy targets
- Preamble to the theory

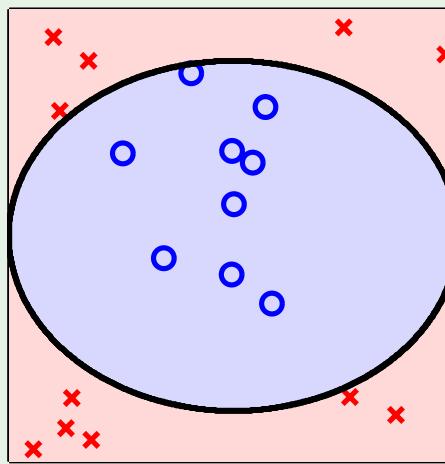


1. Original data
 $\mathbf{x}_n \in \mathcal{X}$

$$\Phi \rightarrow$$

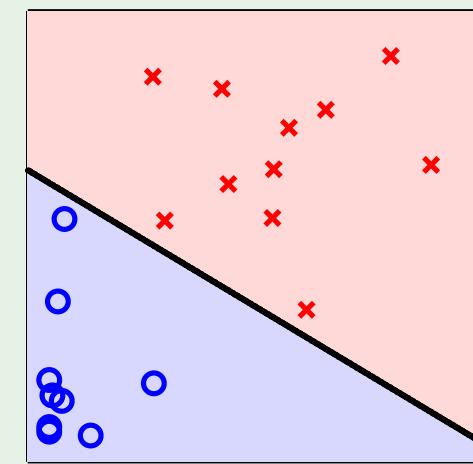


2. Transform the data
 $\mathbf{z}_n = \Phi(\mathbf{x}_n) \in \mathcal{Z}$



4. Classify in \mathcal{X} -space
 $g(\mathbf{x}) = \tilde{g}(\Phi(\mathbf{x})) = \text{sign}(\tilde{\mathbf{w}}^\top \Phi(\mathbf{x}))$

$$\leftarrow \Phi^{-1}$$



3. Separate data in \mathcal{Z} -space
 $\tilde{g}(\mathbf{z}) = \text{sign}(\tilde{\mathbf{w}}^\top \mathbf{z})$

What transforms to what

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

$$\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N \xrightarrow{\Phi} \mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_N$$

$$y_1, y_2, \dots, y_N \xrightarrow{\Phi} y_1, y_2, \dots, y_N$$

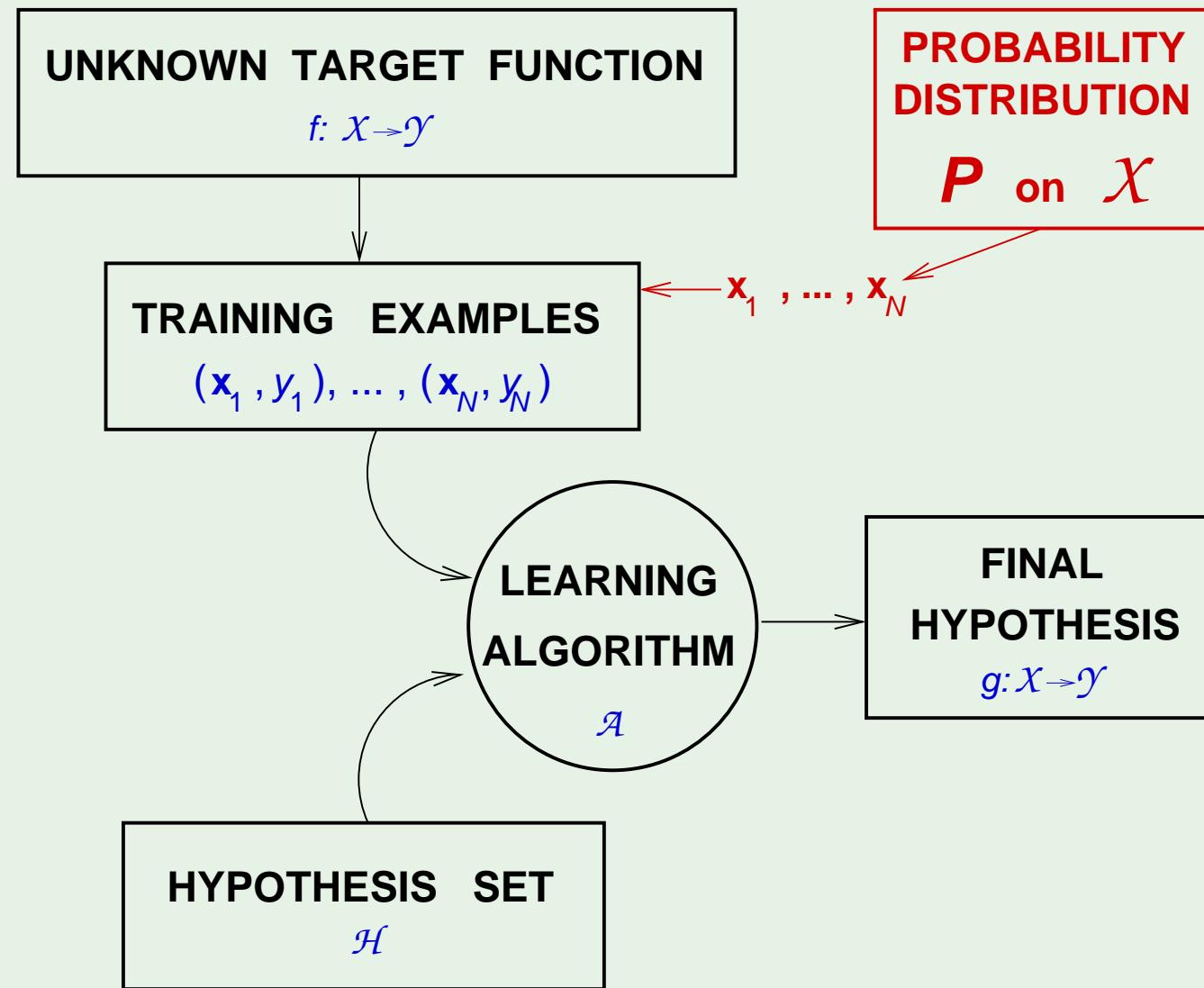
No weights in \mathcal{X} $\tilde{\mathbf{w}} = (w_0, w_1, \dots, w_{\tilde{d}})$

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

Outline

- Nonlinear transformation (continued)
- Error measures
- Noisy targets
- Preamble to the theory

The learning diagram - where we left it



Error measures

What does " $h \approx f$ " mean?

Error measure: $E(h, f)$

Almost always *pointwise definition*: $e(h(\mathbf{x}), f(\mathbf{x}))$

Examples:

Squared error: $e(h(\mathbf{x}), f(\mathbf{x})) = (h(\mathbf{x}) - f(\mathbf{x}))^2$

Binary error: $e(h(\mathbf{x}), f(\mathbf{x})) = \llbracket h(\mathbf{x}) \neq f(\mathbf{x}) \rrbracket$

From pointwise to overall

Overall error $E(h, f)$ = average of pointwise errors $e(h(\mathbf{x}), f(\mathbf{x}))$.

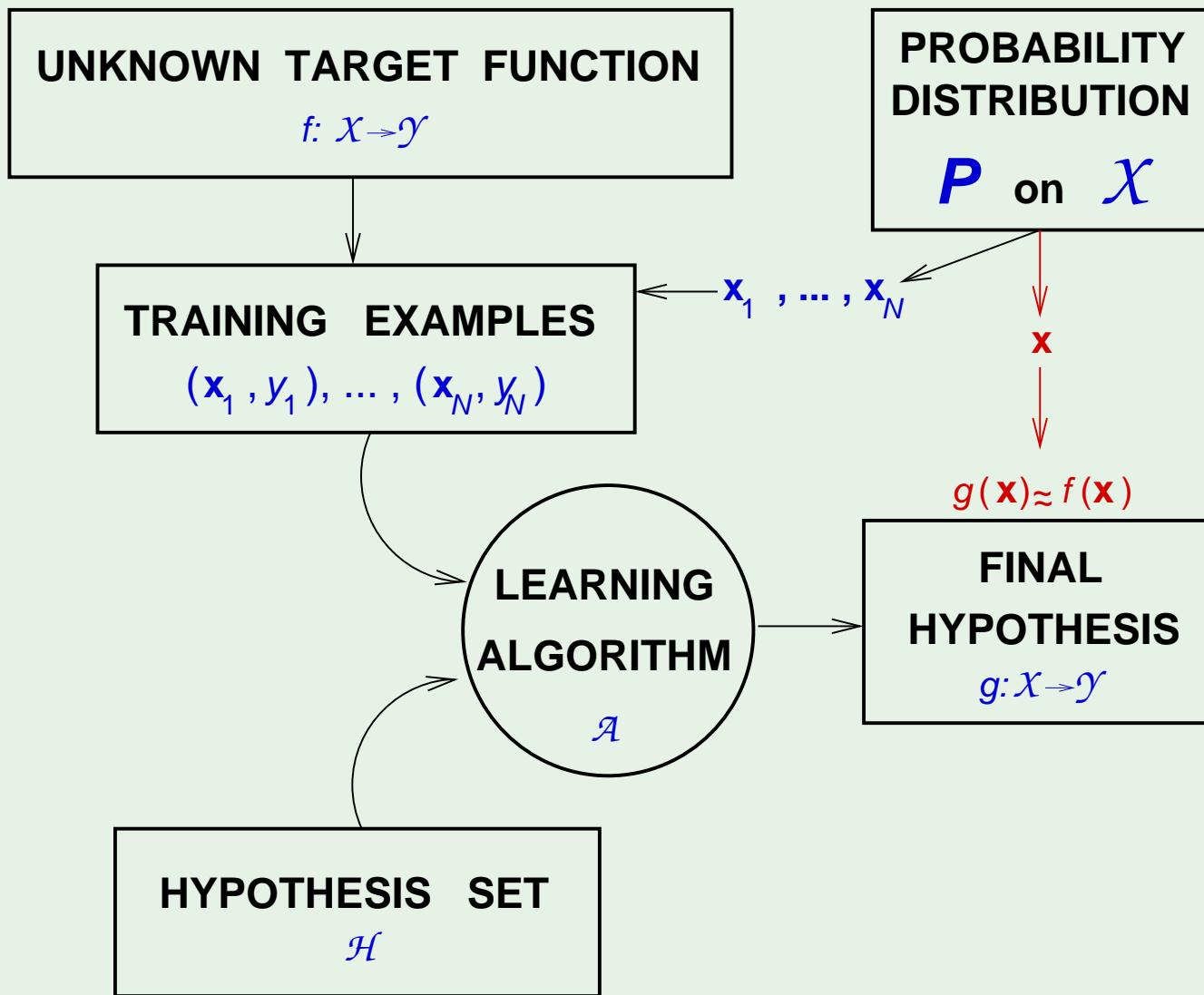
In-sample error:

$$E_{\text{in}}(h) = \frac{1}{N} \sum_{n=1}^N e(h(\mathbf{x}_n), f(\mathbf{x}_n))$$

Out-of-sample error:

$$E_{\text{out}}(h) = \mathbb{E}_{\mathbf{x}} [e(h(\mathbf{x}), f(\mathbf{x}))]$$

The learning diagram - with pointwise error



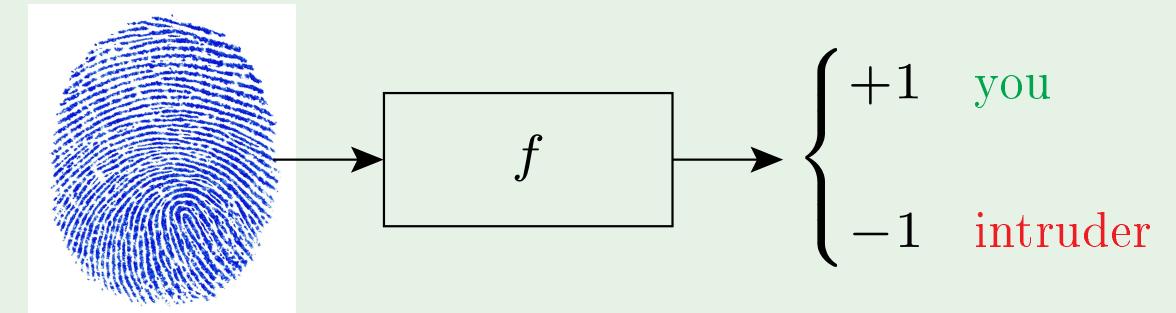
How to choose the error measure

Fingerprint verification:

Two types of error:

false accept and *false reject*

How do we penalize each type?



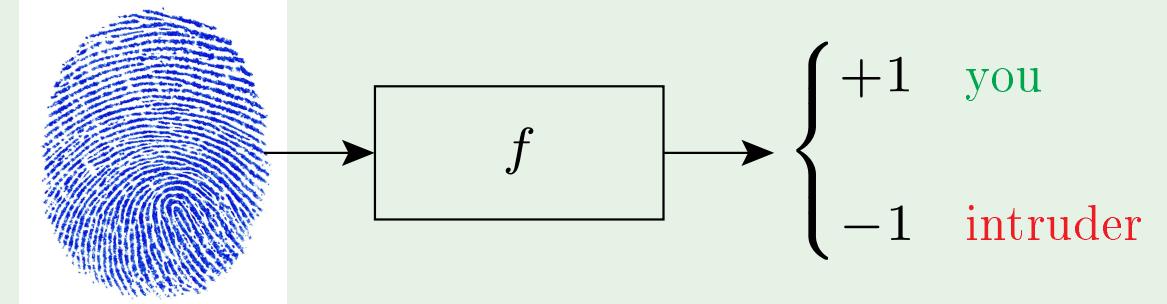
		f	
		+1	-1
h	+1	no error	<i>false accept</i>
	-1	<i>false reject</i>	no error

The error measure - for supermarkets

Supermarket verifies fingerprint for discounts

False reject is costly; customer gets annoyed!

False accept is minor; gave away a discount
and intruder left their fingerprint ☺



		f	
		$+1$	-1
h	$+1$	0	1
	-1	10	0

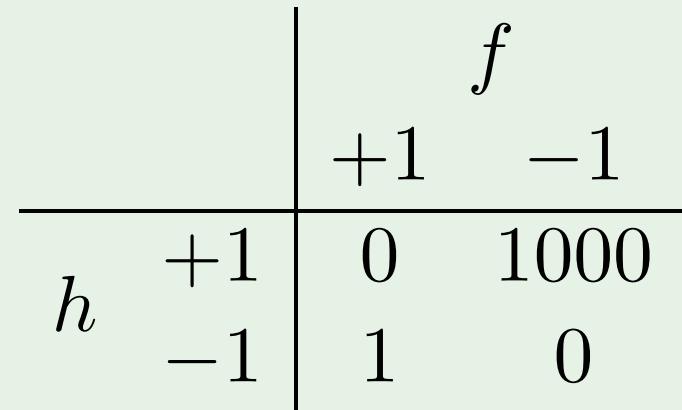
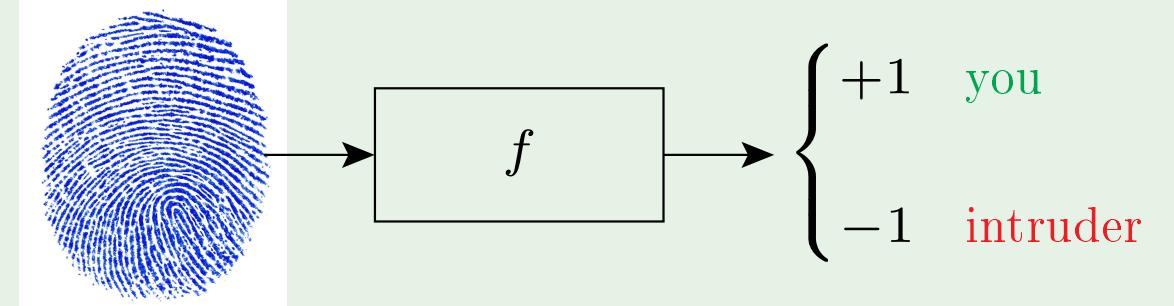
The error measure - for the CIA

CIA verifies fingerprint for security

False accept is a disaster!

False reject can be tolerated

Try again; you are an employee ☺



Take-home lesson

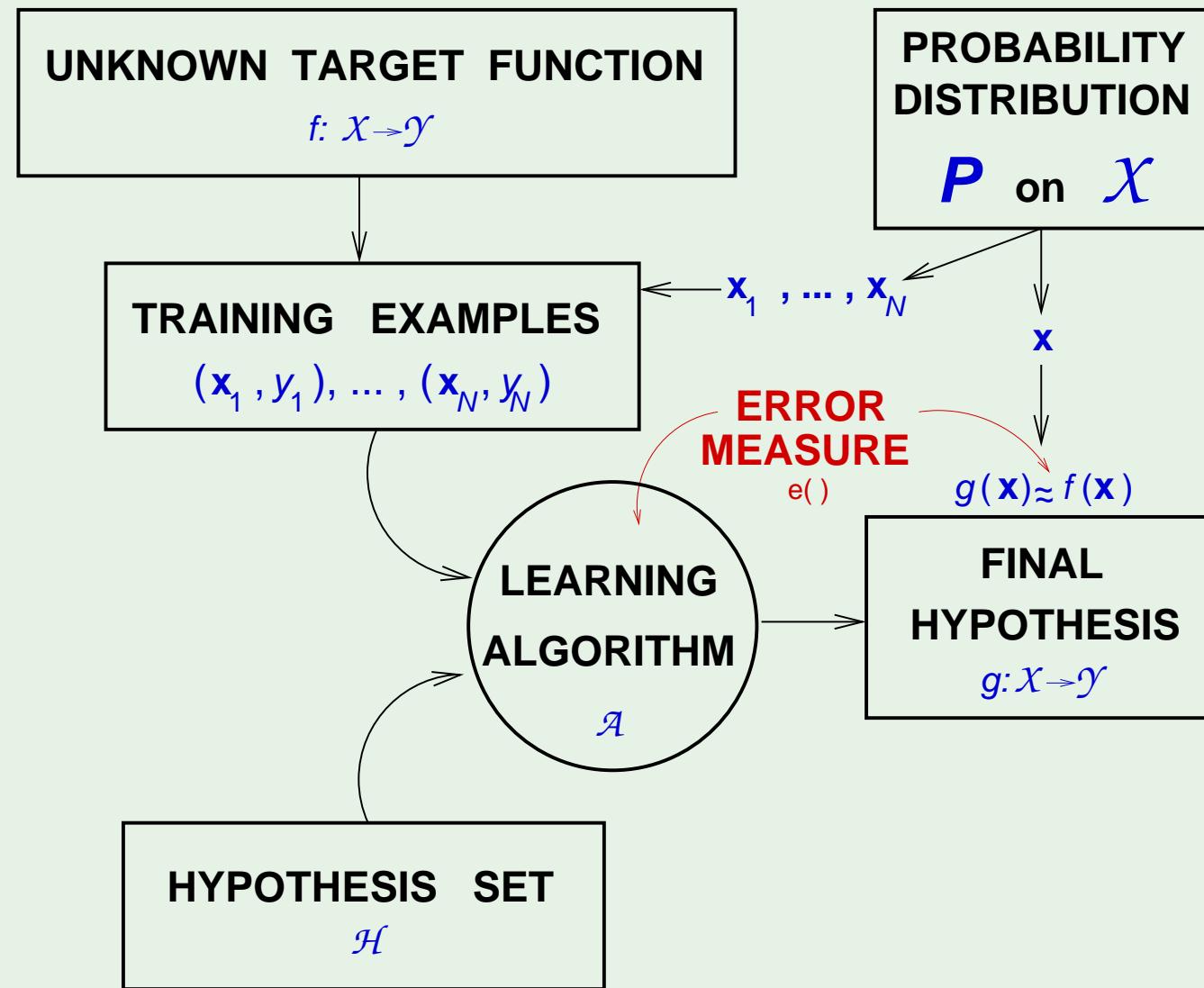
The error measure should be specified by the user.

Not always possible. Alternatives:

Plausible measures: squared error \equiv Gaussian noise

Friendly measures: closed-form solution, convex optimization

The learning diagram - with error measure



Noisy targets

The ‘target function’ is not always a *function*

Consider the credit-card approval:

age	23 years
annual salary	\$30,000
years in residence	1 year
years in job	1 year
current debt	\$15,000
...	...

two ‘identical’ customers → two different behaviors

Target ‘distribution’

Instead of $y = f(\mathbf{x})$, we use target *distribution*:

$$P(y \mid \mathbf{x})$$

(\mathbf{x}, y) is now generated by the joint distribution:

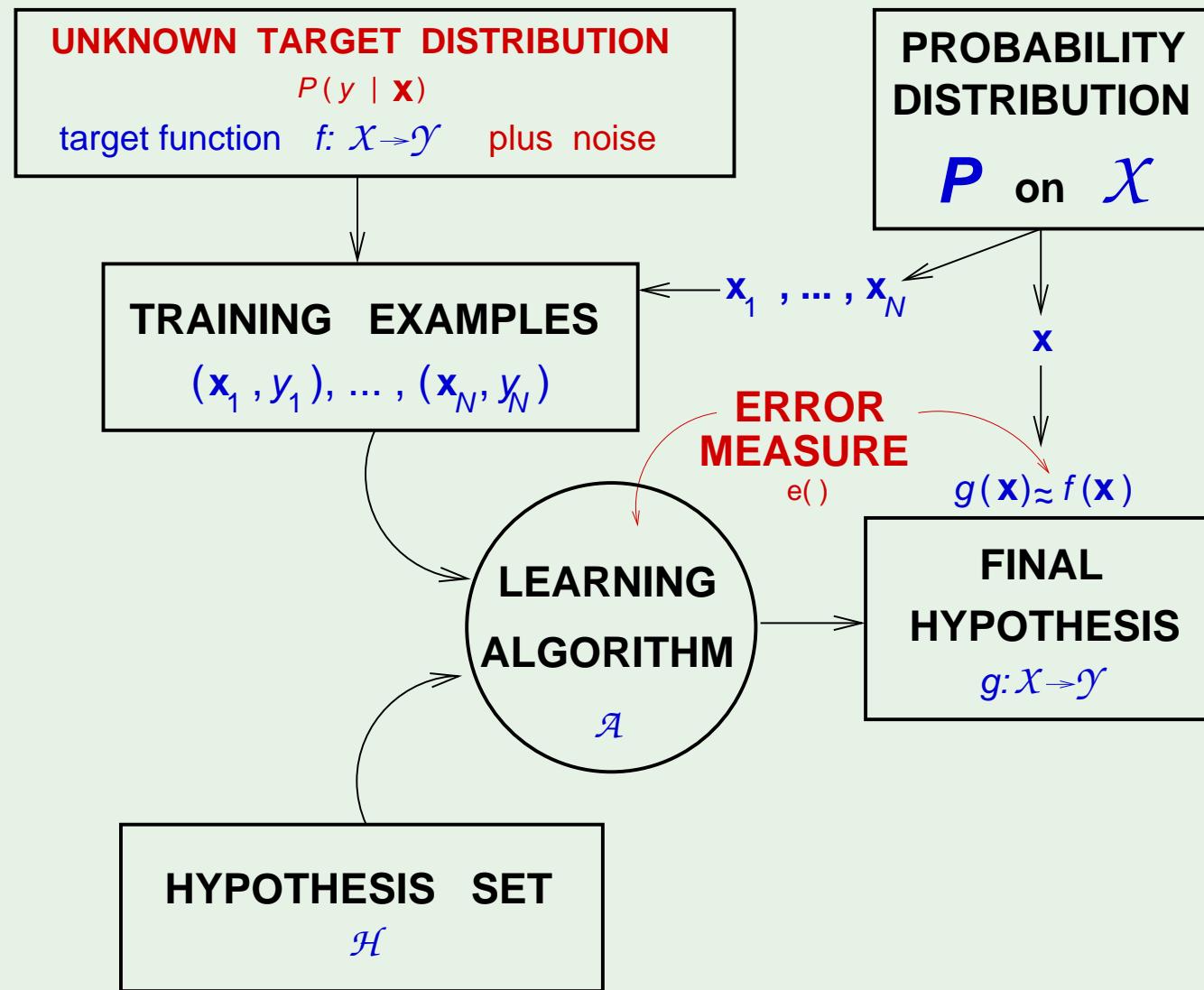
$$P(\mathbf{x})P(y \mid \mathbf{x})$$

Noisy target = deterministic target $f(\mathbf{x}) = \mathbb{E}(y|\mathbf{x})$ plus noise $y - f(\mathbf{x})$

Deterministic target is a special case of noisy target:

$P(y \mid \mathbf{x})$ is zero except for $y = f(\mathbf{x})$

The learning diagram - including noisy target



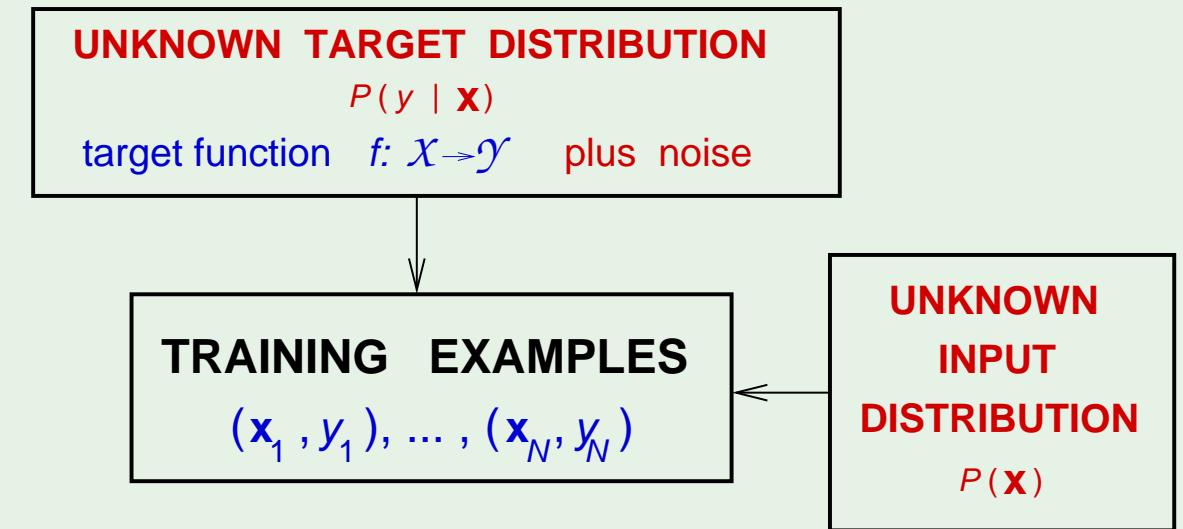
Distinction between $P(y|\mathbf{x})$ and $P(\mathbf{x})$

Both convey probabilistic aspects of \mathbf{x} and y

The target distribution $P(y | \mathbf{x})$
is what we are trying to learn

The input distribution $P(\mathbf{x})$
quantifies relative importance of \mathbf{x}

Merging $P(\mathbf{x})P(y|\mathbf{x})$ as $P(\mathbf{x}, y)$
mixes the two concepts



Outline

- Nonlinear transformation (continued)
- Error measures
- Noisy targets
- Preamble to the theory

What we know so far

Learning is feasible. It is likely that

$$E_{\text{out}}(g) \approx E_{\text{in}}(g)$$

Is this learning?

We need $g \approx f$, which means

$$E_{\text{out}}(g) \approx 0$$

The 2 questions of learning

$E_{\text{out}}(g) \approx 0$ is achieved through:

$$\underbrace{E_{\text{out}}(g) \approx E_{\text{in}}(g)}_{\text{Lecture 2}}$$

and

$$\underbrace{E_{\text{in}}(g) \approx 0}_{\text{Lecture 3}}$$

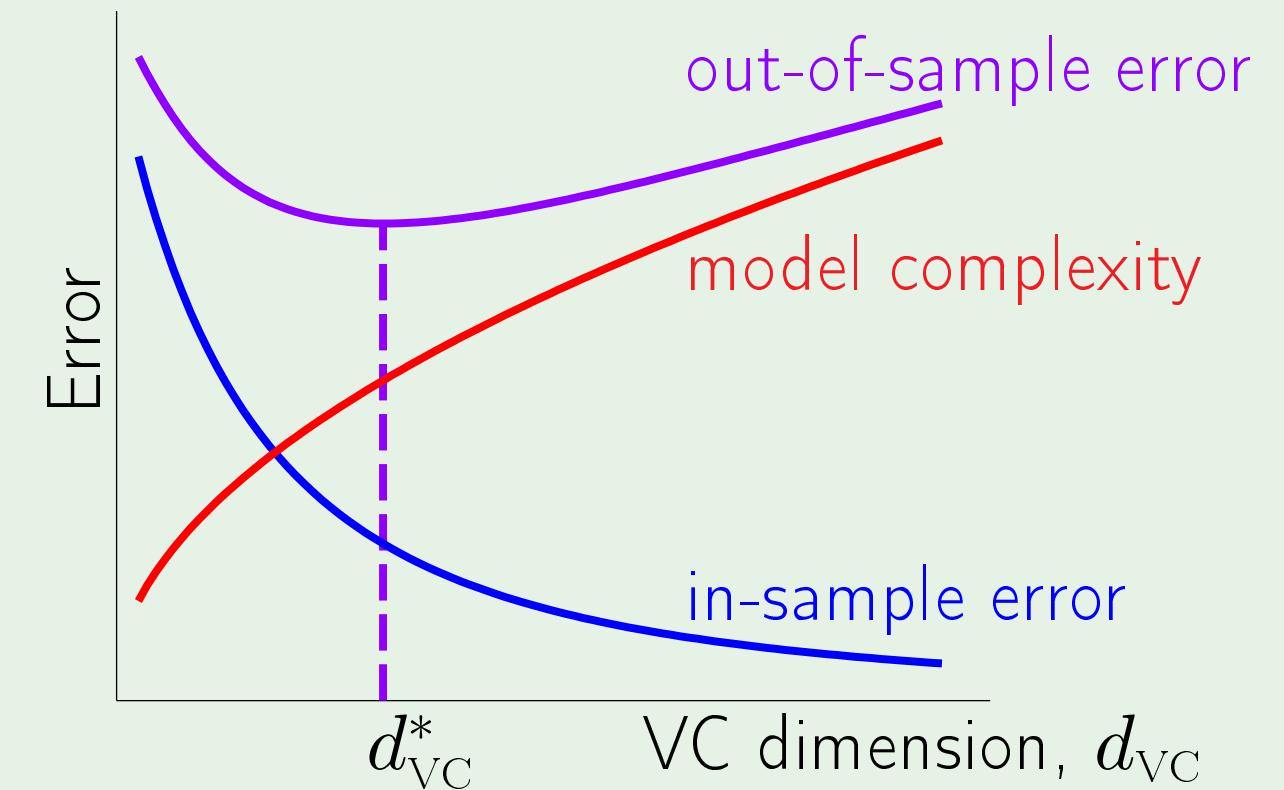
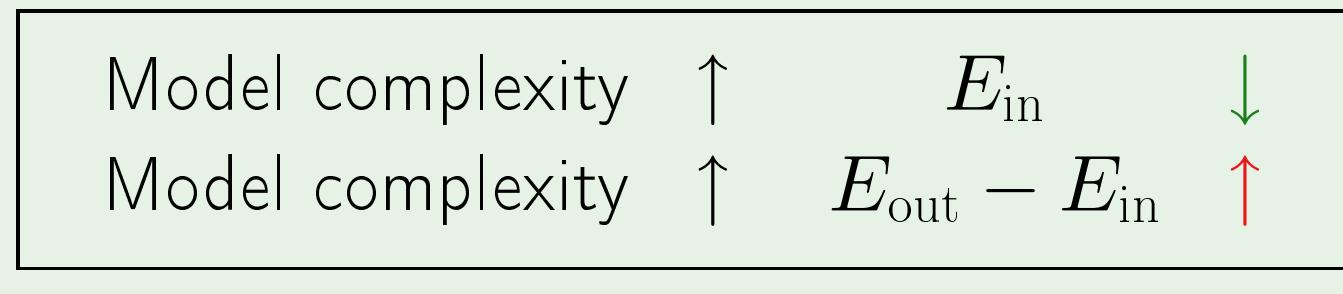
Learning is thus split into 2 questions:

1. Can we make sure that $E_{\text{out}}(g)$ is close enough to $E_{\text{in}}(g)$?
2. Can we make $E_{\text{in}}(g)$ small enough?

What the theory will achieve

Characterizing the feasibility of learning for
infinite M

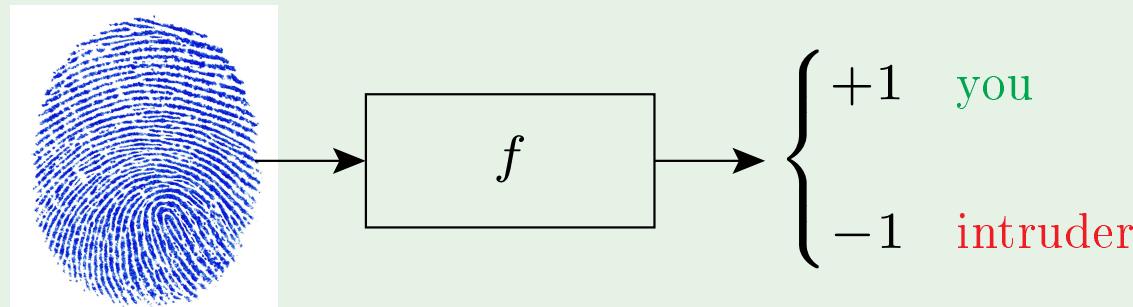
Characterizing the tradeoff:



Review of Lecture 4

- Error measures

- User-specified $e(h(\mathbf{x}), f(\mathbf{x}))$



- In-sample:

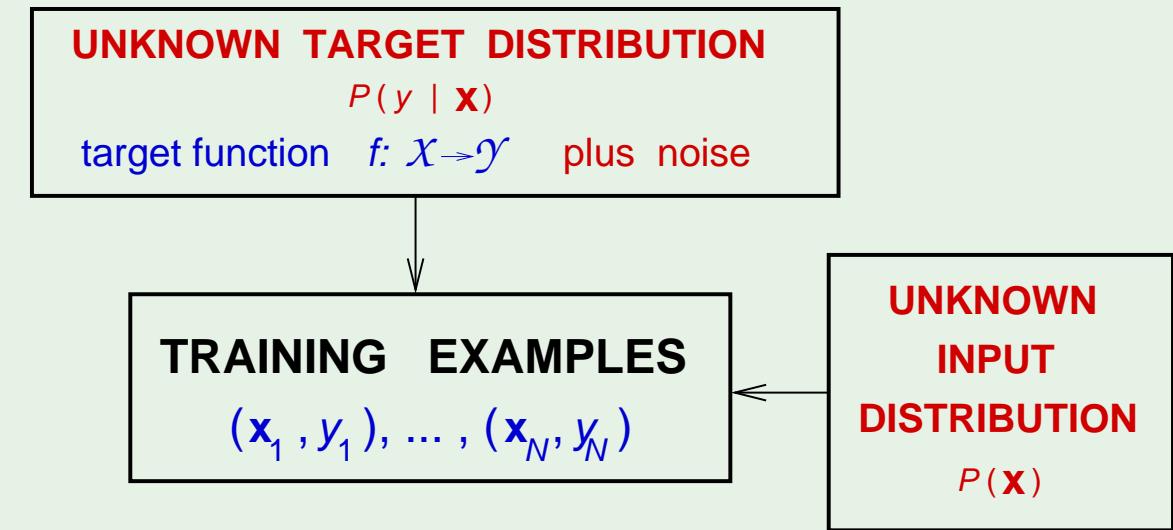
$$E_{\text{in}}(h) = \frac{1}{N} \sum_{n=1}^N e(h(\mathbf{x}_n), f(\mathbf{x}_n))$$

- Out-of-sample

$$E_{\text{out}}(h) = \mathbb{E}_{\mathbf{x}} [e(h(\mathbf{x}), f(\mathbf{x}))]$$

- Noisy targets

$$y = f(\mathbf{x}) \longrightarrow y \sim P(y | \mathbf{x})$$



- $(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)$ generated by

$$P(\mathbf{x}, y) = P(\mathbf{x})P(y|\mathbf{x})$$

- $E_{\text{out}}(h)$ is now $\mathbb{E}_{\mathbf{x}, y} [e(h(\mathbf{x}), y)]$

Learning From Data

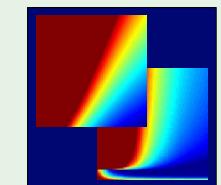
Yaser S. Abu-Mostafa
California Institute of Technology

Lecture 5: Training versus Testing



Sponsored by Caltech's Provost Office, E&AS Division, and IST

• Tuesday, April 17, 2012



Outline

- From training to testing
- Illustrative examples
- Key notion: break point
- Puzzle

The final exam

Testing:

$$\mathbb{P} [|E_{\text{in}} - E_{\text{out}}| > \epsilon] \leq 2 e^{-2\epsilon^2 N}$$

Training:

$$\mathbb{P} [|E_{\text{in}} - E_{\text{out}}| > \epsilon] \leq 2M e^{-2\epsilon^2 N}$$

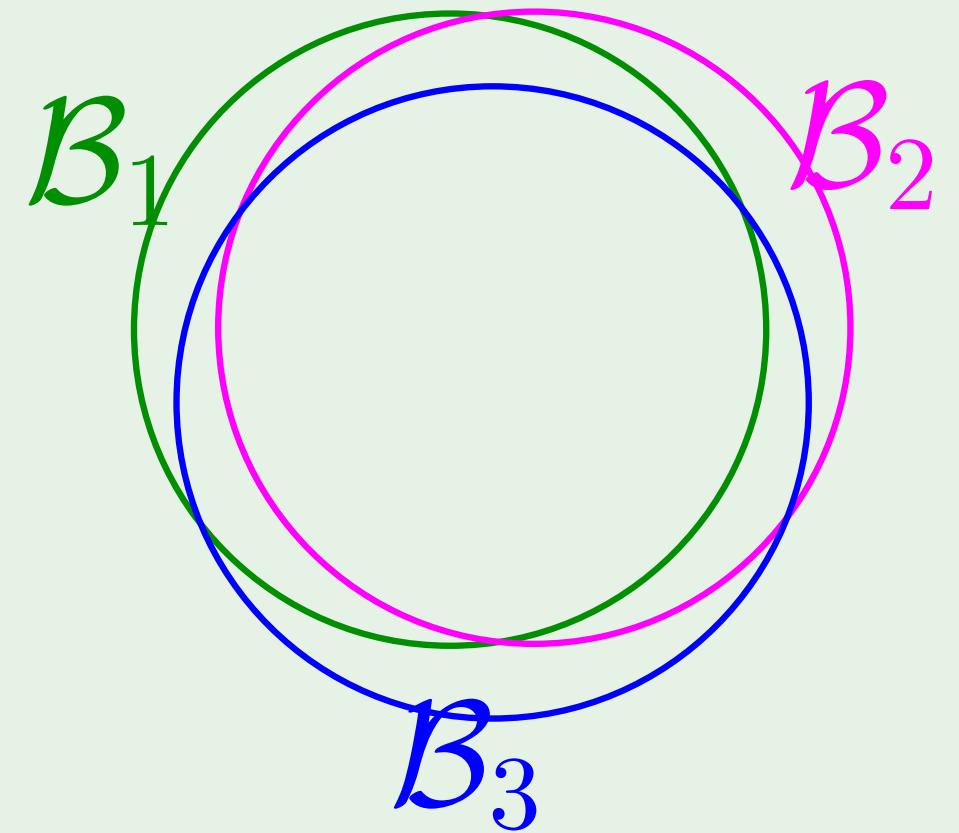
Where did the M come from?

The \mathcal{B} ad events \mathcal{B}_m are

$$“|E_{\text{in}}(h_m) - E_{\text{out}}(h_m)| > \epsilon”$$

The union bound:

$$\begin{aligned} & \mathbb{P}[\mathcal{B}_1 \text{ or } \mathcal{B}_2 \text{ or } \dots \text{ or } \mathcal{B}_M] \\ & \leq \underbrace{\mathbb{P}[\mathcal{B}_1] + \mathbb{P}[\mathcal{B}_2] + \dots + \mathbb{P}[\mathcal{B}_M]}_{\text{no overlaps: } M \text{ terms}} \end{aligned}$$



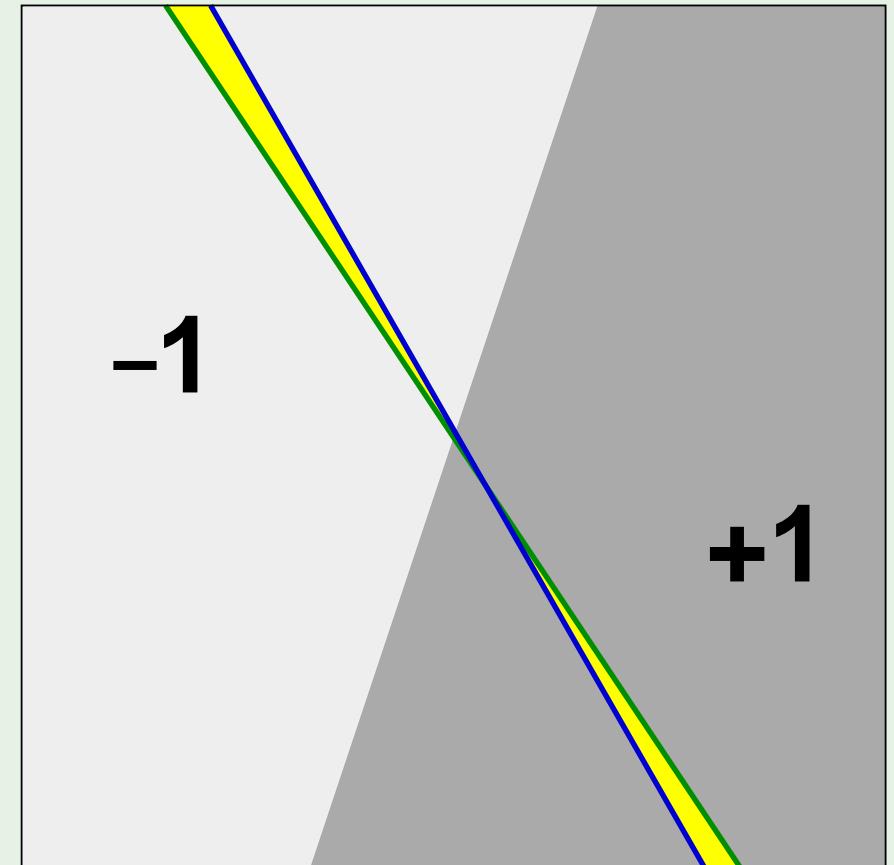
Can we improve on M ?

Yes, bad events are *very* overlapping!

ΔE_{out} : change in $+1$ and -1 areas

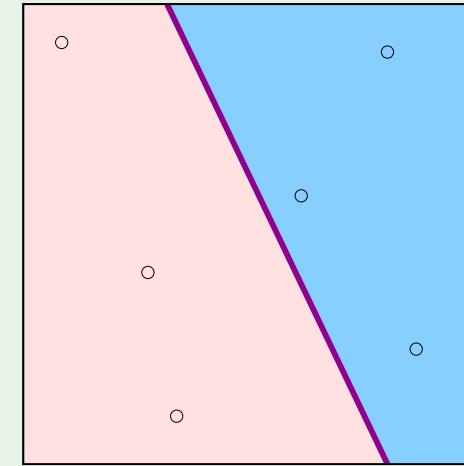
ΔE_{in} : change in labels of data points

$$|E_{\text{in}}(h_1) - E_{\text{out}}(h_1)| \approx |E_{\text{in}}(h_2) - E_{\text{out}}(h_2)|$$

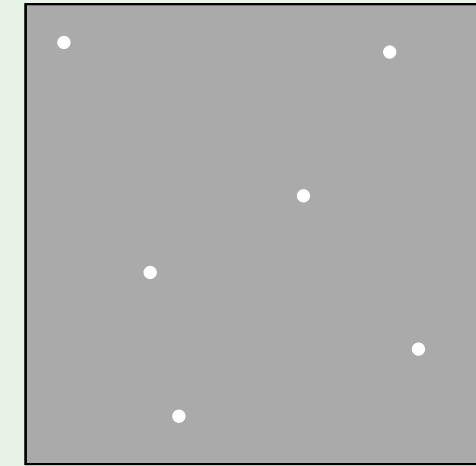


What can we replace M with?

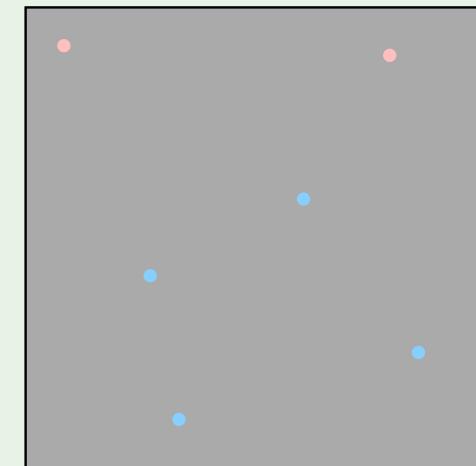
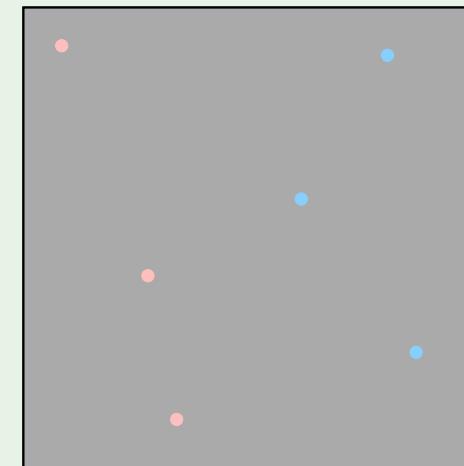
Instead of the whole input space,



we consider a finite set of input points,



and count the number of *dichotomies*



Dichotomies: mini-hypotheses

A hypothesis $h : \mathcal{X} \rightarrow \{-1, +1\}$

A dichotomy $h : \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\} \rightarrow \{-1, +1\}$

Number of hypotheses $|\mathcal{H}|$ can be infinite

Number of dichotomies $|\mathcal{H}(\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N)|$ is at most 2^N

Candidate for replacing M

The growth function

The growth function counts the most dichotomies on any N points

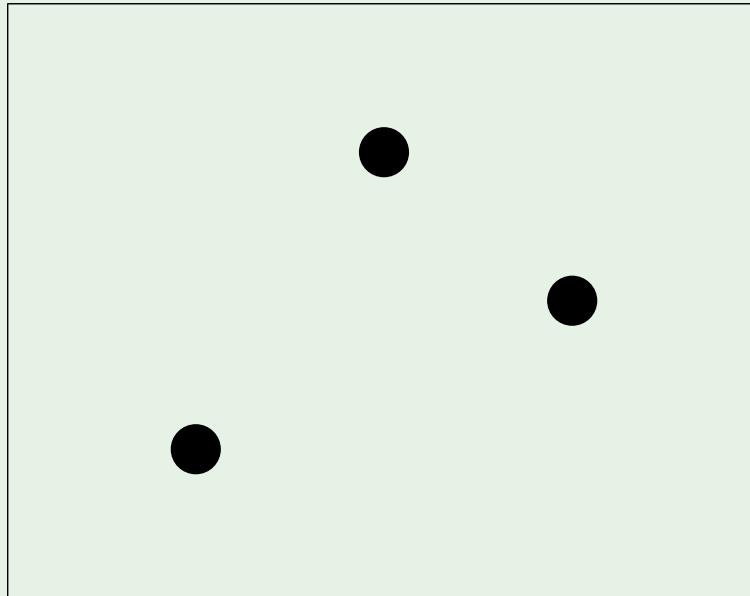
$$m_{\mathcal{H}}(N) = \max_{\mathbf{x}_1, \dots, \mathbf{x}_N \in \mathcal{X}} |\mathcal{H}(\mathbf{x}_1, \dots, \mathbf{x}_N)|$$

The growth function satisfies:

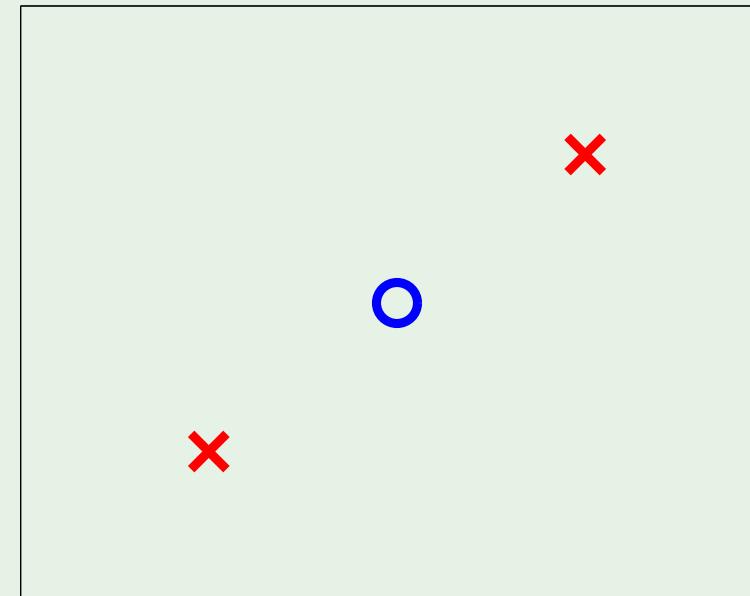
$$m_{\mathcal{H}}(N) \leq 2^N$$

Let's apply the definition.

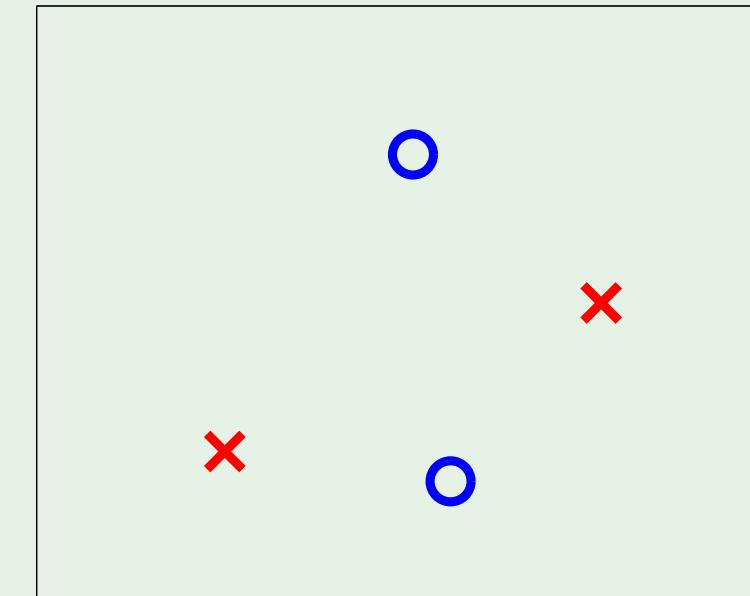
Applying $m_{\mathcal{H}}(N)$ definition - perceptrons



$$N = 3$$



$$N = 3$$



$$N = 4$$

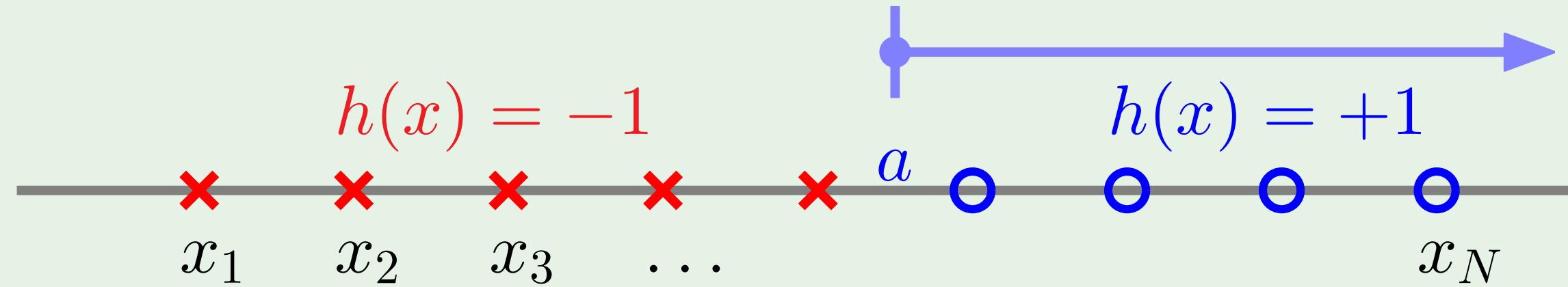
$$m_{\mathcal{H}}(3) = 8$$

$$m_{\mathcal{H}}(4) = 14$$

Outline

- From training to testing
- Illustrative examples
- Key notion: break point
- Puzzle

Example 1: positive rays

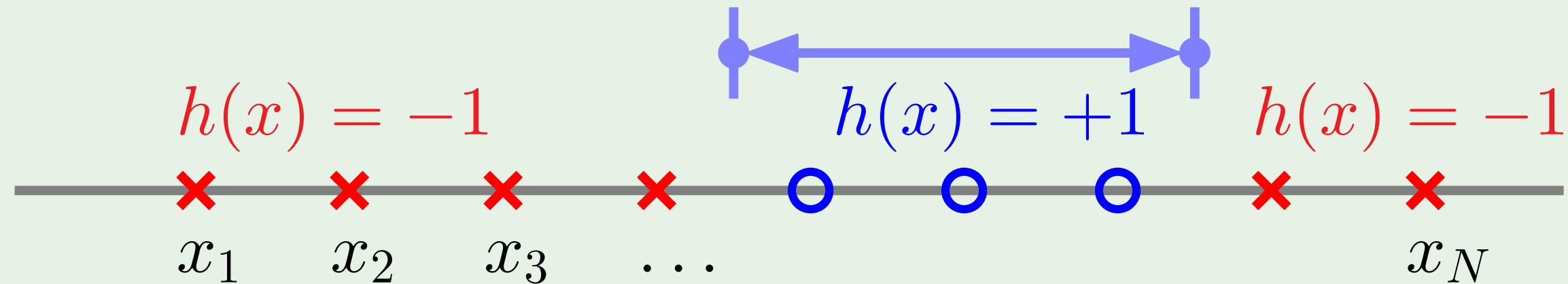


\mathcal{H} is set of $h: \mathbb{R} \rightarrow \{-1, +1\}$

$$h(x) = \text{sign}(x - a)$$

$$\textcolor{red}{m}_{\mathcal{H}}(N) = N + 1$$

Example 2: positive intervals



\mathcal{H} is set of $h: \mathbb{R} \rightarrow \{-1, +1\}$

Place interval ends in two of $N + 1$ spots

$$m_{\mathcal{H}}(N) = \binom{N+1}{2} + 1 = \frac{1}{2}N^2 + \frac{1}{2}N + 1$$

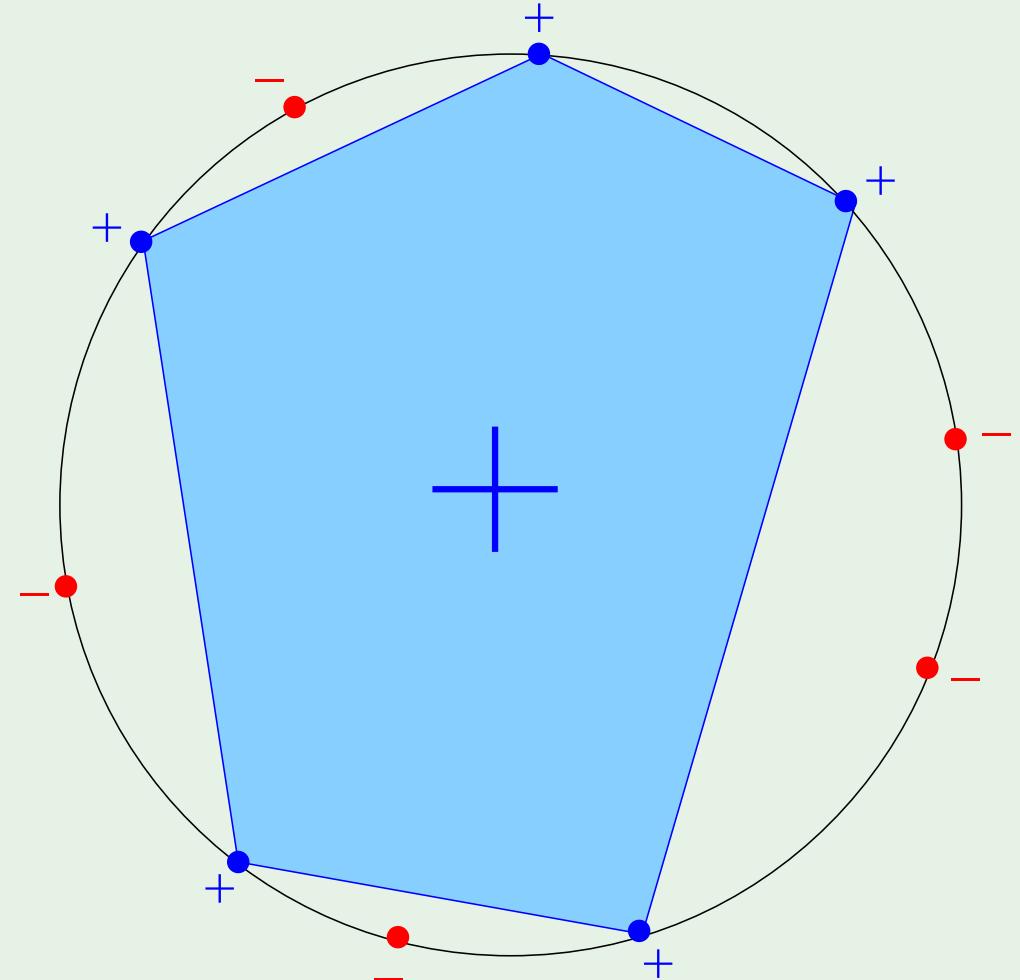
Example 3: convex sets

\mathcal{H} is set of $h: \mathbb{R}^2 \rightarrow \{-1, +1\}$

$h(\mathbf{x}) = +1$ is convex

$m_{\mathcal{H}}(N) = 2^N$

The N points are 'shattered' by convex sets



The 3 growth functions

- \mathcal{H} is positive rays:

$$\textcolor{red}{m}_{\mathcal{H}}(N) = N + 1$$

- \mathcal{H} is positive intervals:

$$\textcolor{red}{m}_{\mathcal{H}}(N) = \frac{1}{2}N^2 + \frac{1}{2}N + 1$$

- \mathcal{H} is convex sets:

$$\textcolor{red}{m}_{\mathcal{H}}(N) = 2^N$$

Back to the big picture

Remember this inequality?

$$\mathbb{P} [|E_{\text{in}} - E_{\text{out}}| > \epsilon] \leq 2M e^{-2\epsilon^2 N}$$

What happens if $m_{\mathcal{H}}(N)$ replaces M ?

$m_{\mathcal{H}}(N)$ polynomial \implies Good!

Just prove that $m_{\mathcal{H}}(N)$ is polynomial?

Outline

- From training to testing
- Illustrative examples
- Key notion: **break point**
- Puzzle

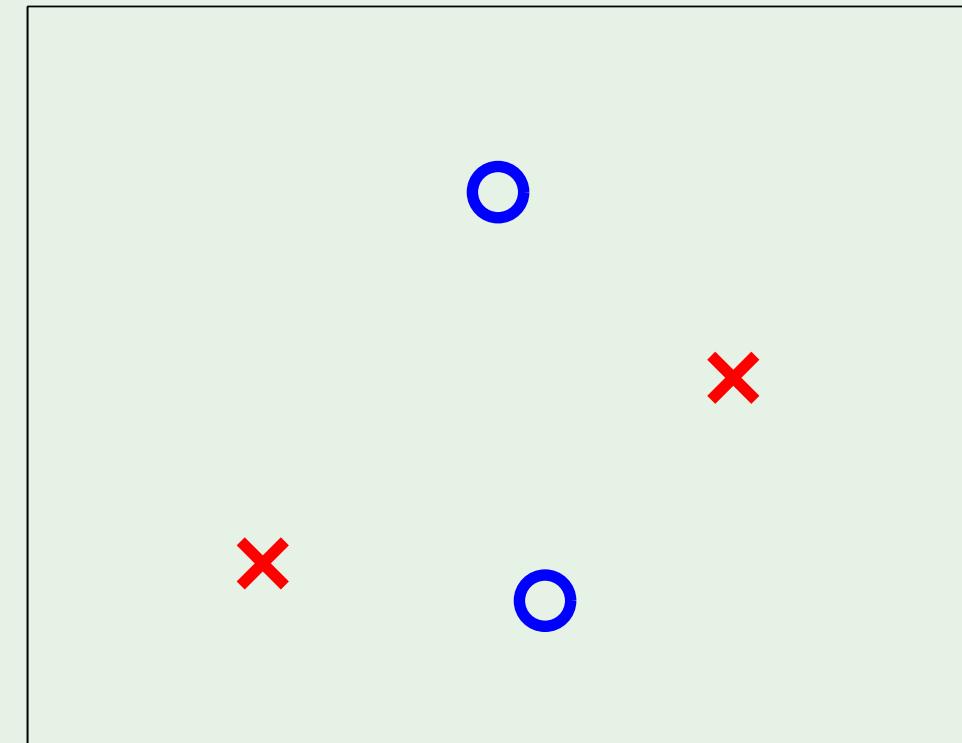
Break point of \mathcal{H}

Definition:

If no data set of size k can be shattered by \mathcal{H} ,
then k is a break point for \mathcal{H}

$$m_{\mathcal{H}}(k) < 2^k$$

For 2D perceptrons, $k = 4$

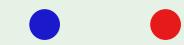


A bigger data set cannot be shattered either

Break point - the 3 examples

- Positive rays $m_{\mathcal{H}}(N) = N + 1$

break point $k = 2$



- Positive intervals $m_{\mathcal{H}}(N) = \frac{1}{2}N^2 + \frac{1}{2}N + 1$

break point $k = 3$



- Convex sets $m_{\mathcal{H}}(N) = 2^N$

break point $k = \infty$

Main result

No break point $\implies m_{\mathcal{H}}(N) = 2^N$

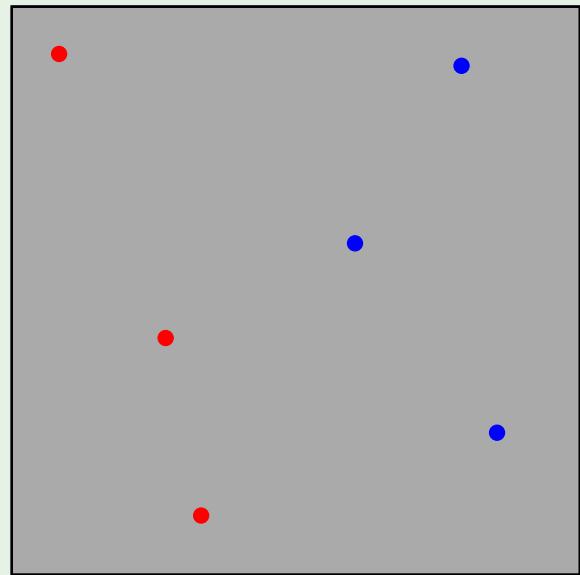
Any break point $\implies m_{\mathcal{H}}(N)$ is polynomial in N

Puzzle

\mathbf{x}_1	\mathbf{x}_2	\mathbf{x}_3
○	○	○
○	○	●
○	●	○
●	○	○

Review of Lecture 5

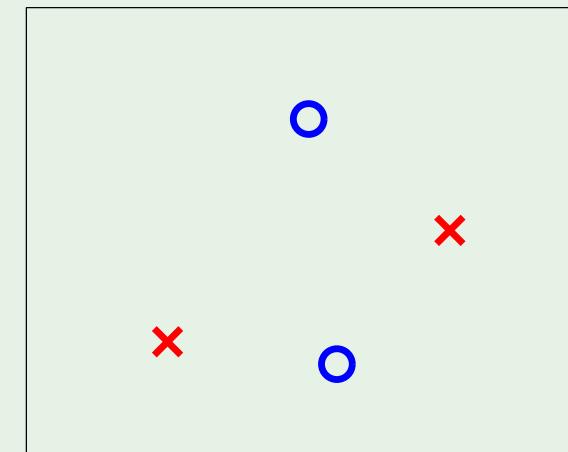
- Dichotomies



- Growth function

$$m_{\mathcal{H}}(N) = \max_{\mathbf{x}_1, \dots, \mathbf{x}_N \in \mathcal{X}} |\mathcal{H}(\mathbf{x}_1, \dots, \mathbf{x}_N)|$$

- Break point



- Maximum # of dichotomies

\mathbf{x}_1	\mathbf{x}_2	\mathbf{x}_3
○	○	○
○	○	●
○	●	○
●	○	○

Learning From Data

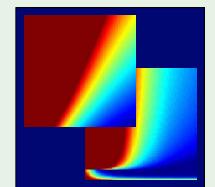
Yaser S. Abu-Mostafa
California Institute of Technology

Lecture 6: Theory of Generalization



Sponsored by Caltech's Provost Office, E&AS Division, and IST

• Thursday, April 19, 2012



Outline

- Proof that $m_{\mathcal{H}}(N)$ is polynomial
- Proof that $m_{\mathcal{H}}(N)$ can replace M

Bounding $m_{\mathcal{H}}(N)$

To show: $m_{\mathcal{H}}(N)$ is polynomial

We show: $m_{\mathcal{H}}(N) \leq \dots \leq \dots \leq$ a polynomial

Key quantity:

$B(N, k)$: Maximum number of dichotomies on N points, with break point k

Recursive bound on $B(N, k)$

Consider the following table:

$$B(N, k) = \alpha + 2\beta$$

	# of rows	\mathbf{x}_1	\mathbf{x}_2	\dots	\mathbf{x}_{N-1}	\mathbf{x}_N
S_1	α	+1	+1	\dots	+1	+1
		-1	+1	\dots	+1	-1
		:	:	:	:	:
		+1	-1	\dots	-1	-1
		-1	+1	\dots	-1	+1
S_2^+	β	+1	-1	\dots	+1	+1
		-1	-1	\dots	+1	+1
		:	:	:	:	:
		+1	-1	\dots	+1	+1
		-1	-1	\dots	-1	+1
S_2^-	β	+1	-1	\dots	+1	-1
		-1	-1	\dots	+1	-1
		:	:	:	:	:
		+1	-1	\dots	+1	-1
		-1	-1	\dots	-1	-1

Estimating α and β

Focus on $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_{N-1}$ columns:

$$\alpha + \beta \leq B(N-1, k)$$

	# of rows	\mathbf{x}_1	\mathbf{x}_2	\dots	\mathbf{x}_{N-1}	\mathbf{x}_N
S_1	α	+1	+1	\dots	+1	+1
		-1	+1	\dots	+1	-1
		:	:	:	:	:
		+1	-1	\dots	-1	-1
		-1	+1	\dots	-1	+1
S_2^+	β	+1	-1	\dots	+1	+1
		-1	-1	\dots	+1	+1
		:	:	:	:	:
		+1	-1	\dots	+1	+1
		-1	-1	\dots	-1	+1
S_2^-	β	+1	-1	\dots	+1	-1
		-1	-1	\dots	+1	-1
		:	:	:	:	:
		+1	-1	\dots	+1	-1
		-1	-1	\dots	-1	-1

Estimating β by itself

Now, focus on the $S_2 = S_2^+ \cup S_2^-$ rows:

$$\beta \leq B(N-1, k-1)$$

	# of rows	\mathbf{x}_1	\mathbf{x}_2	\dots	\mathbf{x}_{N-1}	\mathbf{x}_N
S_1	α	+1	+1	\dots	+1	+1
		-1	+1	\dots	+1	-1
		:	:	\vdots	:	:
		+1	-1	\dots	-1	-1
		-1	+1	\dots	-1	+1
S_2^+	β	+1	-1	\dots	+1	+1
		-1	-1	\dots	+1	+1
		:	:	\vdots	:	:
		+1	-1	\dots	+1	+1
		-1	-1	\dots	-1	+1
S_2^-	β	+1	-1	\dots	+1	-1
		-1	-1	\dots	+1	-1
		:	:	\vdots	:	:
		+1	-1	\dots	+1	-1
		-1	-1	\dots	-1	-1

Putting it together

$$B(N, k) = \alpha + 2\beta$$

$$\alpha + \beta \leq B(N - 1, k)$$

$$\beta \leq B(N - 1, k - 1)$$

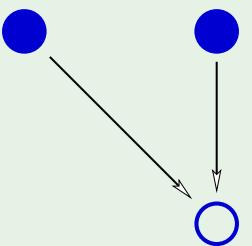
$$B(N, k) \leq$$

$$B(N - 1, k) + B(N - 1, k - 1)$$

	# of rows	\mathbf{x}_1	\mathbf{x}_2	\dots	\mathbf{x}_{N-1}	\mathbf{x}_N
S_1	α	+1	+1	\dots	+1	+1
		-1	+1	\dots	+1	-1
		:	:	:	:	:
		+1	-1	\dots	-1	-1
		-1	+1	\dots	-1	+1
S_2^+	β	+1	-1	\dots	+1	+1
		-1	-1	\dots	+1	+1
		:	:	:	:	:
		+1	-1	\dots	+1	+1
		-1	-1	\dots	-1	+1
S_2^-	β	+1	-1	\dots	+1	-1
		-1	-1	\dots	+1	-1
		:	:	:	:	:
		+1	-1	\dots	+1	-1
		-1	-1	\dots	-1	-1

Numerical computation of $B(N, k)$ bound

$$B(N, k) \leq B(N - 1, k) + B(N - 1, k - 1)$$



		k						
		1	2	3	4	5	6	\dots
N	1	1	2	2	2	2	2	\dots
	2	1	3	4	4	4	4	\dots
	3	1	4	7	8	8	8	\dots
	4	1	5	11	\dots	\dots	\dots	\dots
	5	1	6	\vdots	\ddots			
	6	1	7	\vdots				
	\vdots	\vdots	\vdots	\vdots				

Analytic solution for $B(N, k)$ bound

$$B(N, k) \leq B(N - 1, k) + B(N - 1, k - 1)$$

Theorem:

$$B(N, k) \leq \sum_{i=0}^{k-1} \binom{N}{i}$$

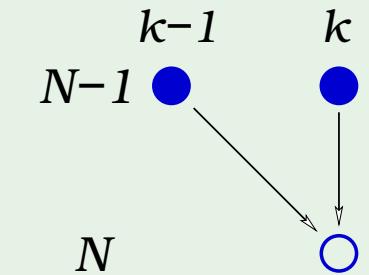
1. Boundary conditions: easy

	k						
	1	2	3	4	5	6	\dots
1	1	2	2	2	2	2	\dots
2	1						
3	1						
N	1						
4	1						
5	1						
6	1						
:	:						

A diagram illustrating a step function or a path. It consists of a grid where the horizontal axis is labeled k and the vertical axis is labeled N . The grid has columns for $k = 1, 2, 3, 4, 5, 6, \dots$ and rows for $N = 1, 2, 3, 4, 5, 6, \dots$. The values in the grid are: row 1: 1, 2, 2, 2, 2, 2, ...; row 2: 1; row 3: 1; row 4: 1; row 5: 1; row 6: 1; and so on. A solid blue dot is placed at the intersection of the fourth column and the fourth row (labeled (4, 1)). From this point, a diagonal arrow points down and to the right, leading to an open circle at the intersection of the fifth column and the fourth row (labeled (5, 1)).

2. The induction step

$$\begin{aligned}
 \sum_{i=0}^{k-1} \binom{N}{i} &= \sum_{i=0}^{k-1} \binom{N-1}{i} + \sum_{i=0}^{k-2} \binom{N-1}{i} ? \\
 &= 1 + \sum_{i=1}^{k-1} \binom{N-1}{i} + \sum_{i=1}^{k-1} \binom{N-1}{i-1} \\
 &= 1 + \sum_{i=1}^{k-1} \left[\binom{N-1}{i} + \binom{N-1}{i-1} \right] \\
 &= 1 + \sum_{i=1}^{k-1} \binom{N}{i} = \sum_{i=0}^{k-1} \binom{N}{i} \checkmark
 \end{aligned}$$



It is polynomial!

For a given \mathcal{H} , the break point k is fixed

$$m_{\mathcal{H}}(N) \leq \underbrace{\sum_{i=0}^{k-1} \binom{N}{i}}_{\text{maximum power is } N^{k-1}}$$

Three examples

$$\sum_{i=0}^{k-1} \binom{N}{i}$$

- \mathcal{H} is **positive rays**: (break point $k = 2$)

$$m_{\mathcal{H}}(N) = N + 1 \leq N + 1$$

- \mathcal{H} is **positive intervals**: (break point $k = 3$)

$$m_{\mathcal{H}}(N) = \frac{1}{2}N^2 + \frac{1}{2}N + 1 \leq \frac{1}{2}N^2 + \frac{1}{2}N + 1$$

- \mathcal{H} is **2D perceptrons**: (break point $k = 4$)

$$m_{\mathcal{H}}(N) = ? \leq \frac{1}{6}N^3 + \frac{5}{6}N + 1$$

Outline

- Proof that $m_{\mathcal{H}}(N)$ is polynomial
- Proof that $m_{\mathcal{H}}(N)$ can replace M

What we want

Instead of:

$$\mathbb{P}[|E_{\text{in}}(g) - E_{\text{out}}(g)| > \epsilon] \leq 2 \quad M \quad e^{-2\epsilon^2 N}$$

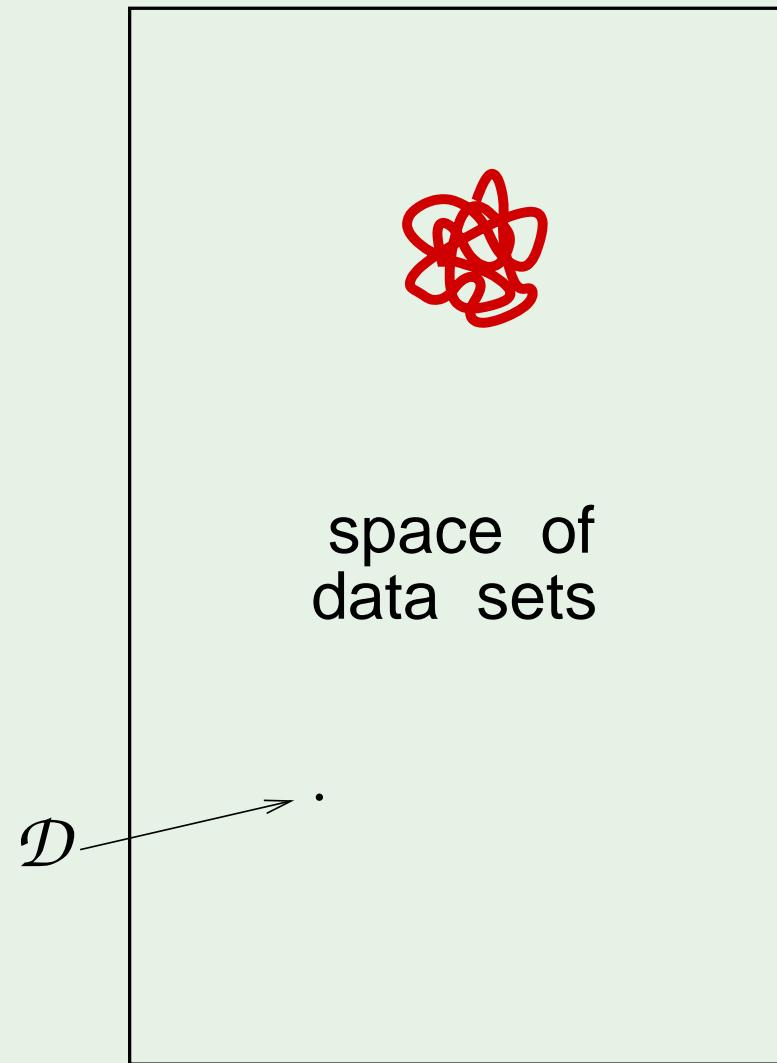
We want:

$$\mathbb{P}[|E_{\text{in}}(g) - E_{\text{out}}(g)| > \epsilon] \leq 2 \quad m_{\mathcal{H}}(N) \quad e^{-2\epsilon^2 N}$$

Pictorial proof 😊

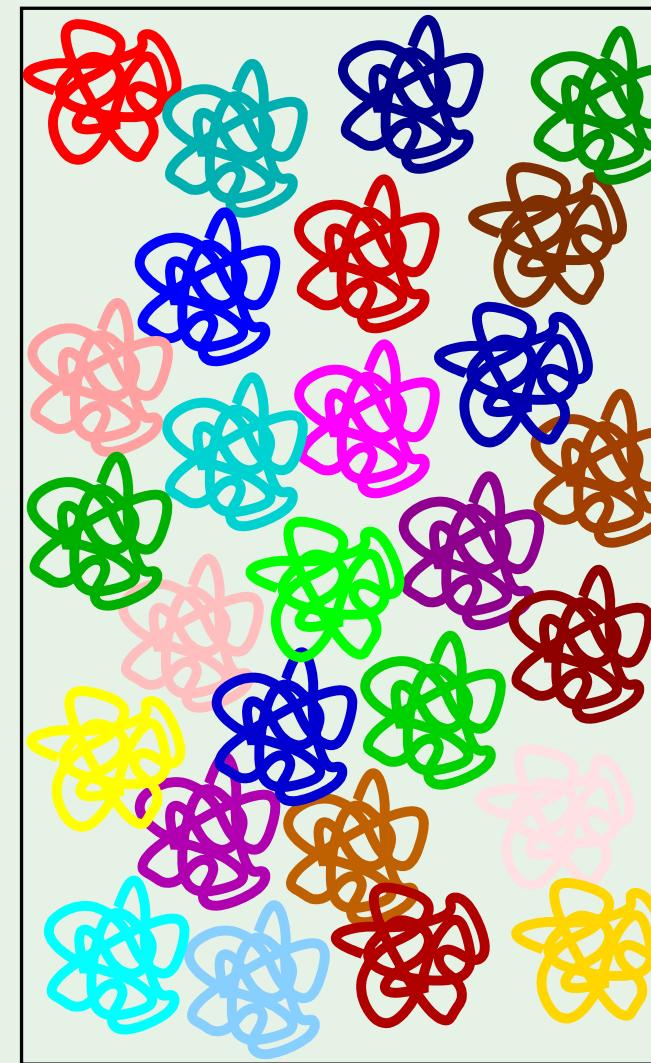
- How does $m_{\mathcal{H}}(N)$ relate to overlaps?
- What to do about E_{out} ?
- Putting it together

Hoeffding Inequality



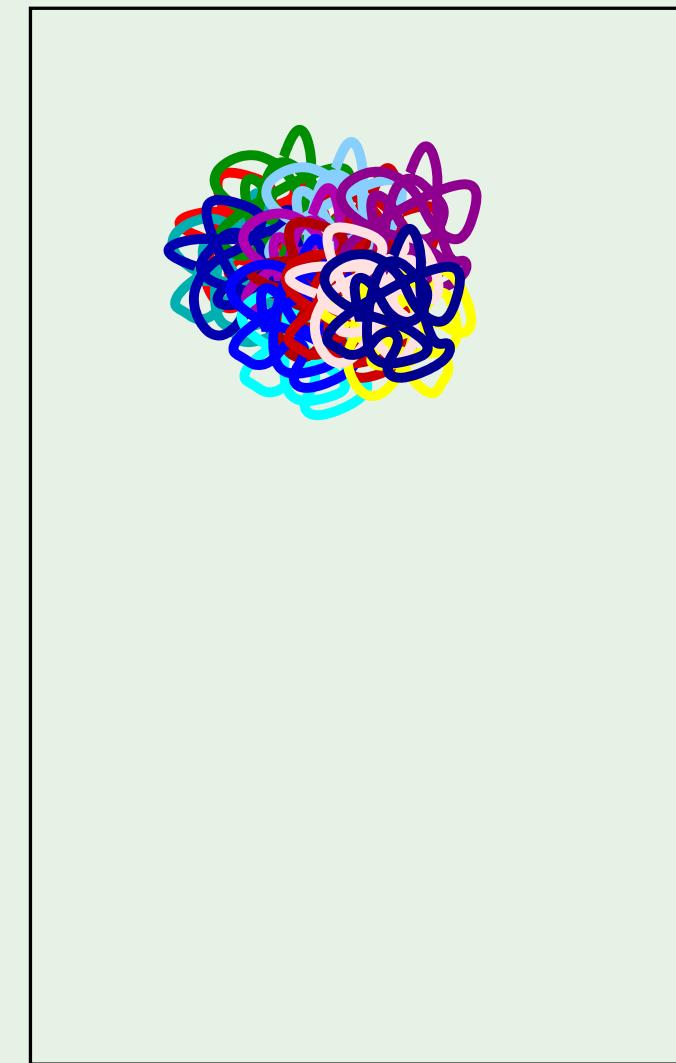
(a)

Union Bound



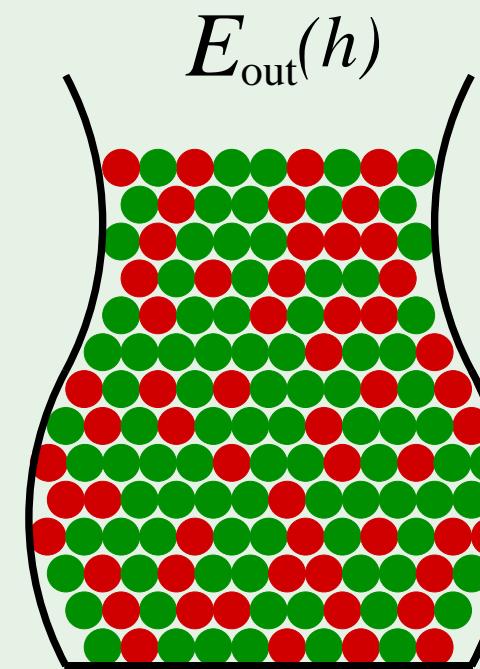
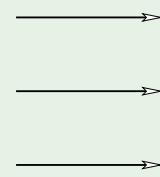
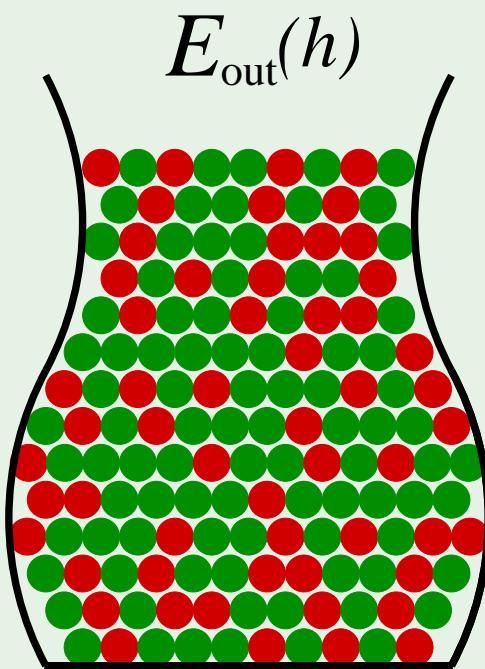
(b)

VC Bound



(c)

What to do about E_{out}



● ● ● ● ● ● ●

$E_{\text{in}}(h)$

● ● ● ● ● ● ●

$E'_{\text{in}}(h)$ ● ● ● ● ● ● ●

Putting it together

Not quite:

$$\mathbb{P}[|E_{\text{in}}(g) - E_{\text{out}}(g)| > \epsilon] \leq 2 m_{\mathcal{H}}(N) e^{-2\epsilon^2 N}$$

but rather:

$$\mathbb{P}[|E_{\text{in}}(g) - E_{\text{out}}(g)| > \epsilon] \leq 4 m_{\mathcal{H}}(2N) e^{-\frac{1}{8}\epsilon^2 N}$$

The Vapnik-Chervonenkis Inequality

Review of Lecture 6

- $m_{\mathcal{H}}(N)$ is polynomial

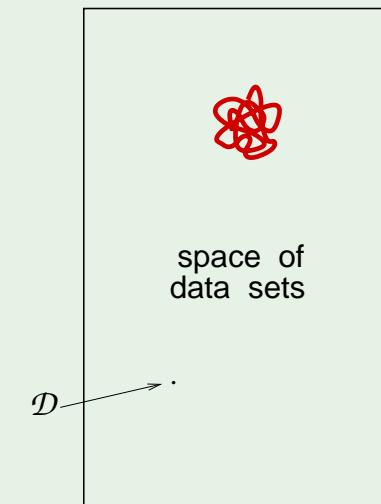
if \mathcal{H} has a break point k

		1	2	3	4	5	6	\dots
	1	1	2	2	2	2	2	\dots
	2		1					
	3			1				
N	4				1			
	5					1		
	6						1	
	:							

$$m_{\mathcal{H}}(N) \leq \underbrace{\sum_{i=0}^{k-1} \binom{N}{i}}_{\text{maximum power is } N^{k-1}}$$

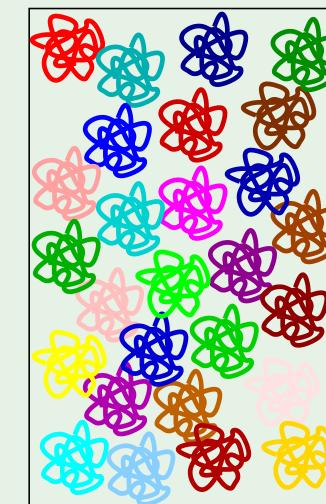
- The VC Inequality

Hoeffding Inequality



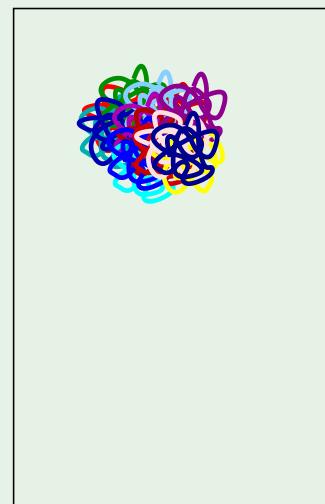
(a)

Union Bound



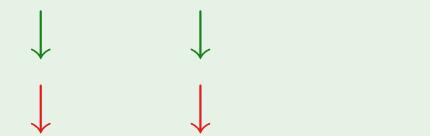
(b)

VC Bound



(c)

$$\mathbb{P} [|E_{\text{in}}(g) - E_{\text{out}}(g)| > \epsilon] \leq 2^M e^{-2\epsilon^2 N}$$



$$\mathbb{P} [|E_{\text{in}}(g) - E_{\text{out}}(g)| > \epsilon] \leq 4 m_{\mathcal{H}}(2N) e^{-\frac{1}{8}\epsilon^2 N}$$

Learning From Data

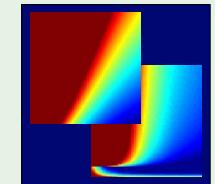
Yaser S. Abu-Mostafa
California Institute of Technology

Lecture 7: The VC Dimension



Sponsored by Caltech's Provost Office, E&AS Division, and IST

• Tuesday, April 24, 2012



Outline

- The definition
- VC dimension of perceptrons
- Interpreting the VC dimension
- Generalization bounds

Definition of VC dimension

The VC dimension of a hypothesis set \mathcal{H} , denoted by $d_{\text{VC}}(\mathcal{H})$, is

the largest value of N for which $m_{\mathcal{H}}(N) = 2^N$

“the most points \mathcal{H} can shatter”

$N \leq d_{\text{VC}}(\mathcal{H}) \implies \mathcal{H}$ can shatter N points

$k > d_{\text{VC}}(\mathcal{H}) \implies k$ is a break point for \mathcal{H}

The growth function

In terms of a break point k :

$$m_{\mathcal{H}}(N) \leq \sum_{i=0}^{k-1} \binom{N}{i}$$

In terms of the VC dimension d_{VC} :

$$m_{\mathcal{H}}(N) \leq \underbrace{\sum_{i=0}^{d_{\text{VC}}} \binom{N}{i}}_{\text{maximum power is } N^{d_{\text{VC}}}}$$

Examples

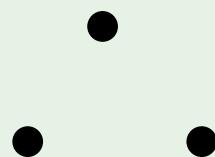
- \mathcal{H} is positive rays:

$$d_{\text{VC}} = 1$$



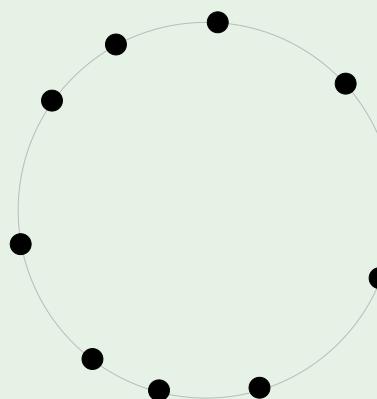
- \mathcal{H} is 2D perceptrons:

$$d_{\text{VC}} = 3$$



- \mathcal{H} is convex sets:

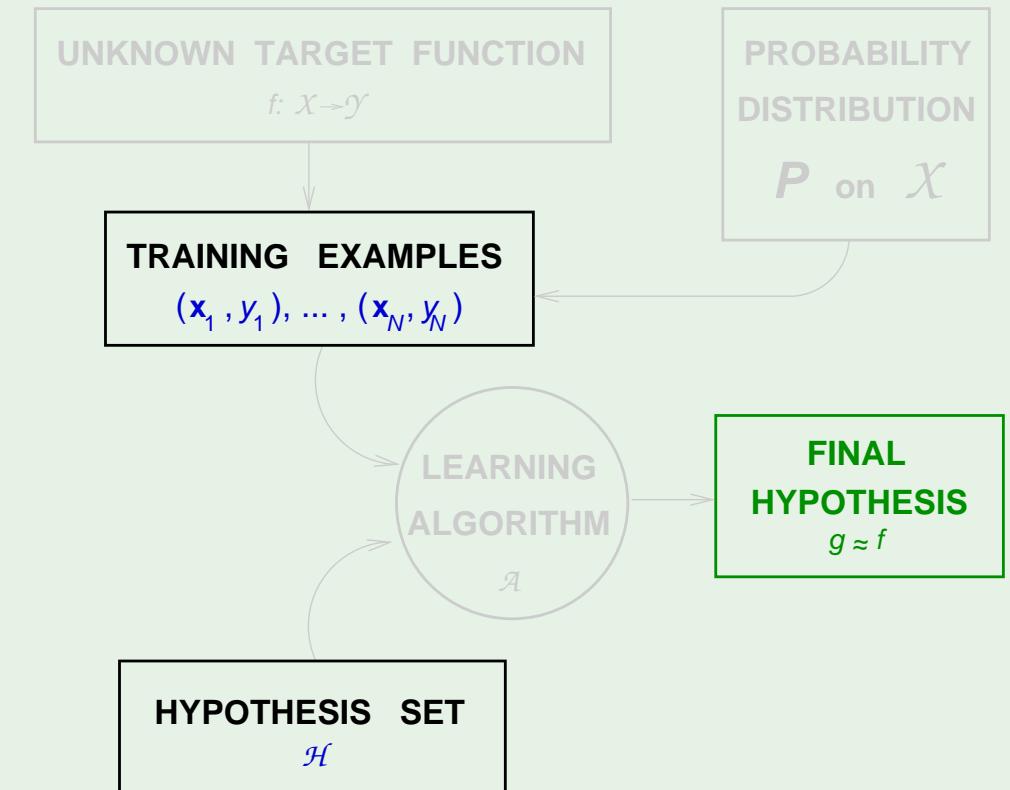
$$d_{\text{VC}} = \infty$$



VC dimension and learning

$d_{\text{VC}}(\mathcal{H})$ is finite $\implies g \in \mathcal{H}$ will generalize

- Independent of the **learning algorithm**
- Independent of the **input distribution**
- Independent of the **target function**



VC dimension of perceptrons

For $d = 2$, $d_{\text{VC}} = 3$

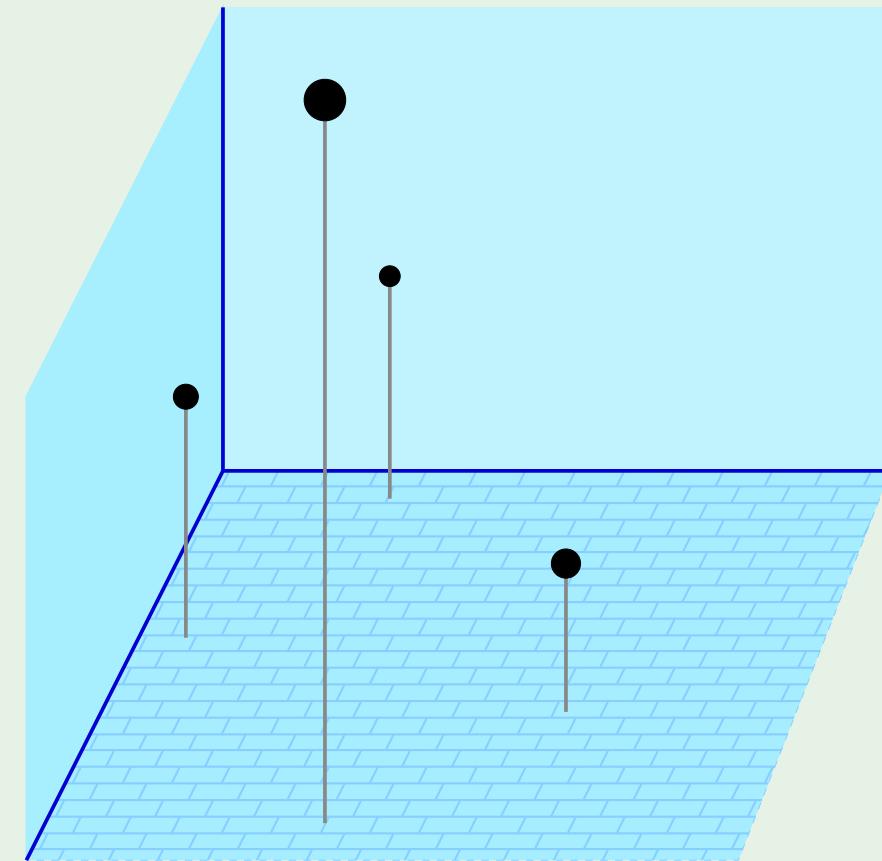
In general,

$d_{\text{VC}} = d + 1$

We will prove two directions:

$$d_{\text{VC}} \leq d + 1$$

$$d_{\text{VC}} \geq d + 1$$



Here is one direction

A set of $N = d + 1$ points in \mathbb{R}^d shattered by the perceptron:

$$X = \begin{bmatrix} \mathbf{x}_1^\top \\ \mathbf{x}_2^\top \\ \mathbf{x}_3^\top \\ \vdots \\ \mathbf{x}_{d+1}^\top \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ 1 & 1 & 0 & \dots & 0 \\ 1 & 0 & 1 & & 0 \\ \vdots & & & \ddots & 0 \\ 1 & 0 & \dots & 0 & 1 \end{bmatrix}$$

X is invertible

Can we shatter this data set?

For any $\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{d+1} \end{bmatrix} = \begin{bmatrix} \pm 1 \\ \pm 1 \\ \vdots \\ \pm 1 \end{bmatrix}$, can we find a vector \mathbf{w} satisfying

$$\text{sign}(\mathbf{X}\mathbf{w}) = \mathbf{y}$$

Easy! Just make $\mathbf{X}\mathbf{w} = \mathbf{y}$

which means $\mathbf{w} = \mathbf{X}^{-1}\mathbf{y}$

We can shatter these $d + 1$ points

This implies what?

[a] $d_{\text{VC}} = d + 1$

[b] $d_{\text{VC}} \geq d + 1$ ✓

[c] $d_{\text{VC}} \leq d + 1$

[d] No conclusion

Now, to show that $d_{VC} \leq d + 1$

We need to show that:

- [a] There are $d + 1$ points we cannot shatter
- [b] There are $d + 2$ points we cannot shatter
- [c] We cannot shatter *any* set of $d + 1$ points
- [d] We cannot shatter *any* set of $d + 2$ points ✓

Take any $d + 2$ points

For any $d + 2$ points,

$$\mathbf{x}_1, \dots, \mathbf{x}_{d+1}, \mathbf{x}_{d+2}$$

More points than dimensions \implies we must have

$$\mathbf{x}_j = \sum_{i \neq j} \mathbf{a}_i \mathbf{x}_i$$

where not all the \mathbf{a}_i 's are zeros

So?

$$\mathbf{x}_j = \sum_{i \neq j} \textcolor{red}{a_i} \mathbf{x}_i$$

Consider the following dichotomy:

\mathbf{x}_i 's with non-zero $\textcolor{red}{a_i}$ get $y_i = \text{sign}(\textcolor{red}{a_i})$

and \mathbf{x}_j gets $y_j = -1$

No perceptron can implement such dichotomy!

Why?

$$\mathbf{x}_j = \sum_{i \neq j} a_i \mathbf{x}_i \implies \mathbf{w}^\top \mathbf{x}_j = \sum_{i \neq j} a_i \mathbf{w}^\top \mathbf{x}_i$$

If $y_j = \text{sign}(\mathbf{w}^\top \mathbf{x}_j) = \text{sign}(a_j)$, then $a_j \mathbf{w}^\top \mathbf{x}_j > 0$

This forces

$$\mathbf{w}^\top \mathbf{x}_j = \sum_{i \neq j} a_i \mathbf{w}^\top \mathbf{x}_i > 0$$

Therefore, $y_j = \text{sign}(\mathbf{w}^\top \mathbf{x}_j) = +1$

Putting it together

We proved $d_{\text{VC}} \leq d + 1$ and $d_{\text{VC}} \geq d + 1$

$$d_{\text{VC}} = d + 1$$

What is $d + 1$ in the perceptron?

It is the number of parameters w_0, w_1, \dots, w_d

Outline

- The definition
- VC dimension of perceptrons
- Interpreting the VC dimension
- Generalization bounds

1. Degrees of freedom

Parameters create degrees of freedom

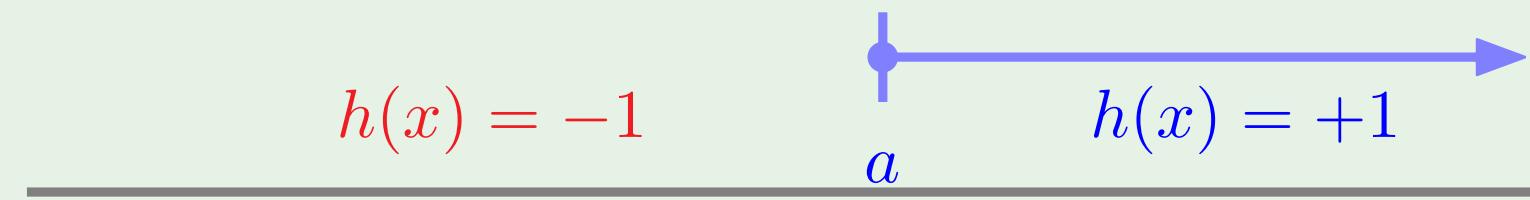
of parameters: **analog** degrees of freedom

d_{VC} : equivalent ‘**binary**’ degrees of freedom

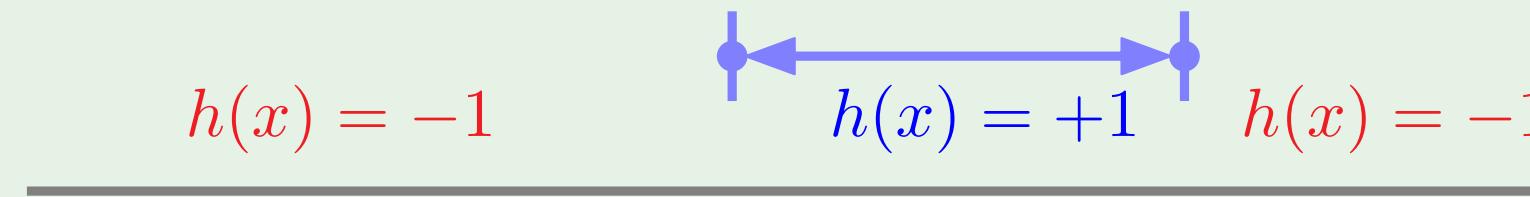


The usual suspects

Positive rays ($d_{VC} = 1$):

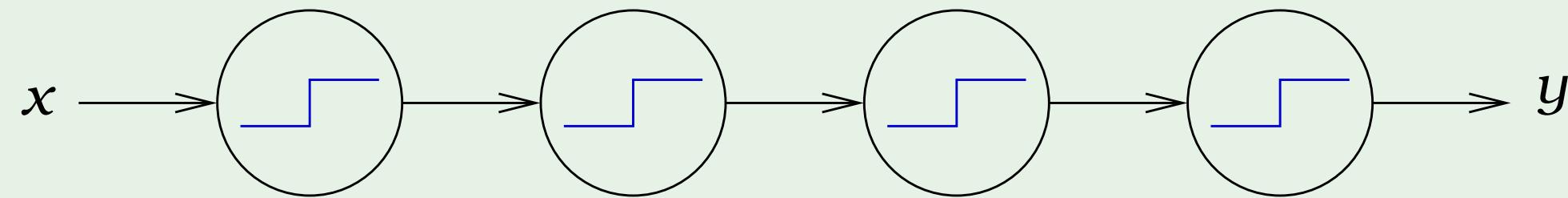


Positive intervals ($d_{VC} = 2$):



Not just parameters

Parameters may not contribute degrees of freedom:



d_{VC} measures the **effective** number of parameters

2. Number of data points needed

Two small quantities in the VC inequality:

$$\mathbb{P}[|E_{\text{in}}(g) - E_{\text{out}}(g)| > \epsilon] \leq \underbrace{4m_{\mathcal{H}}(2N)e^{-\frac{1}{8}\epsilon^2 N}}_{\delta}$$

If we want certain ϵ and δ , how does N depend on d_{VC} ?

Let us look at

$$N^d e^{-N}$$

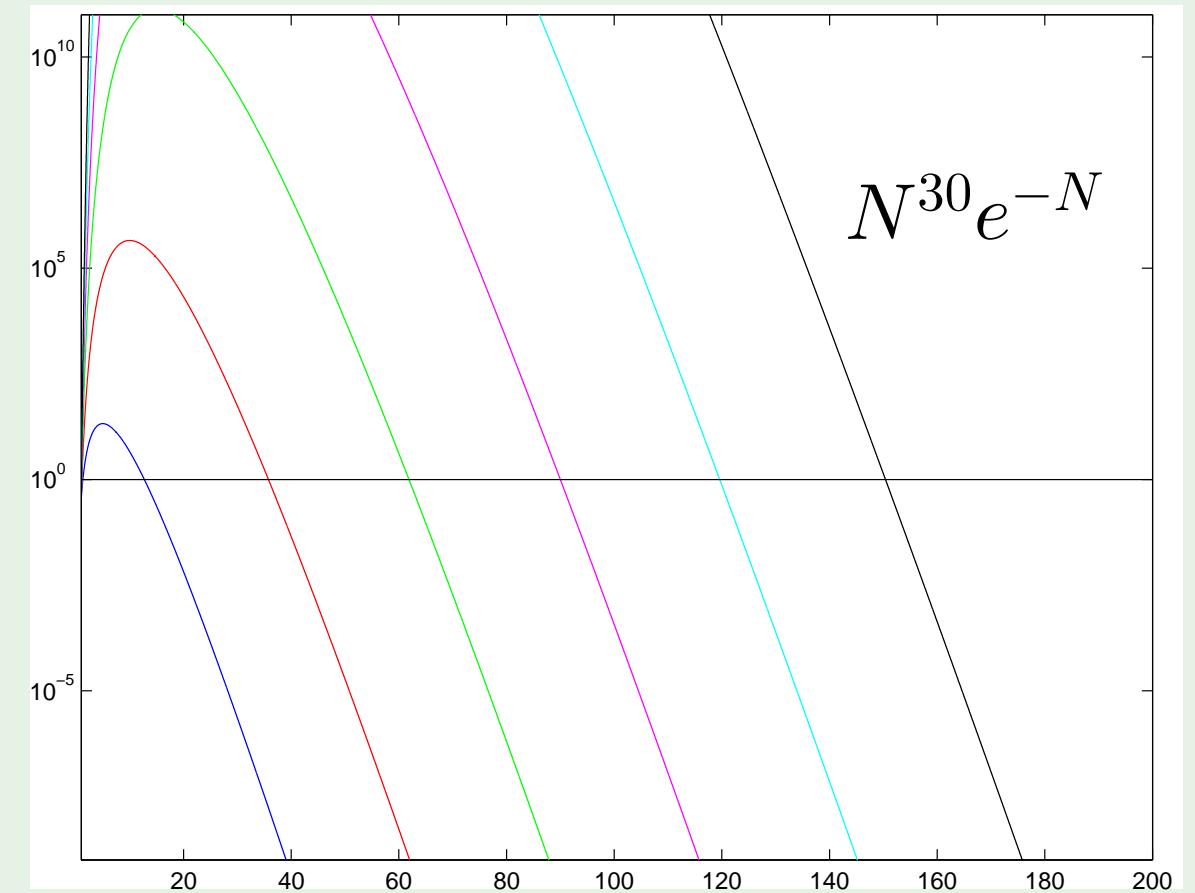
$$N^{\textcolor{red}{d}} e^{-N}$$

Fix $N^{\textcolor{red}{d}} e^{-N} = \text{small value}$

How does N change with $\textcolor{red}{d}$?

Rule of thumb:

$$N \geq 10 \textcolor{red}{d}_{\text{VC}}$$



Outline

- The definition
- VC dimension of perceptrons
- Interpreting the VC dimension
- Generalization bounds

Rearranging things

Start from the VC inequality:

$$\mathbb{P}[|E_{\text{out}} - E_{\text{in}}| > \epsilon] \leq \underbrace{4m_{\mathcal{H}}(2N)e^{-\frac{1}{8}\epsilon^2 N}}_{\delta}$$

Get ϵ in terms of δ :

$$\delta = 4m_{\mathcal{H}}(2N)e^{-\frac{1}{8}\epsilon^2 N} \implies \epsilon = \sqrt{\underbrace{\frac{8}{N} \ln \frac{4m_{\mathcal{H}}(2N)}{\delta}}_{\Omega}}$$

With probability $\geq 1 - \delta$, $|E_{\text{out}} - E_{\text{in}}| \leq \Omega(N, \mathcal{H}, \delta)$

Generalization bound

With probability $\geq 1 - \delta$,

$$E_{\text{out}} - E_{\text{in}} \leq \Omega$$

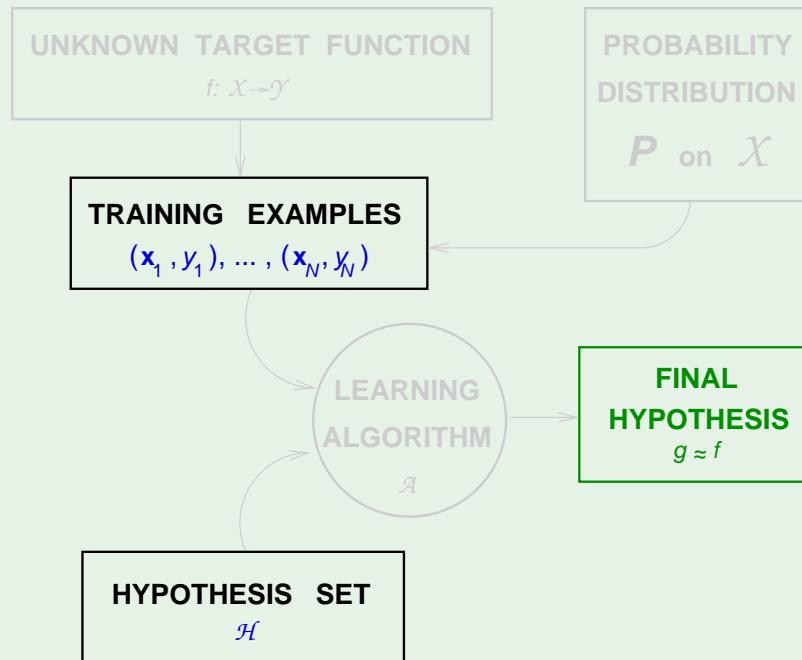


With probability $\geq 1 - \delta$,

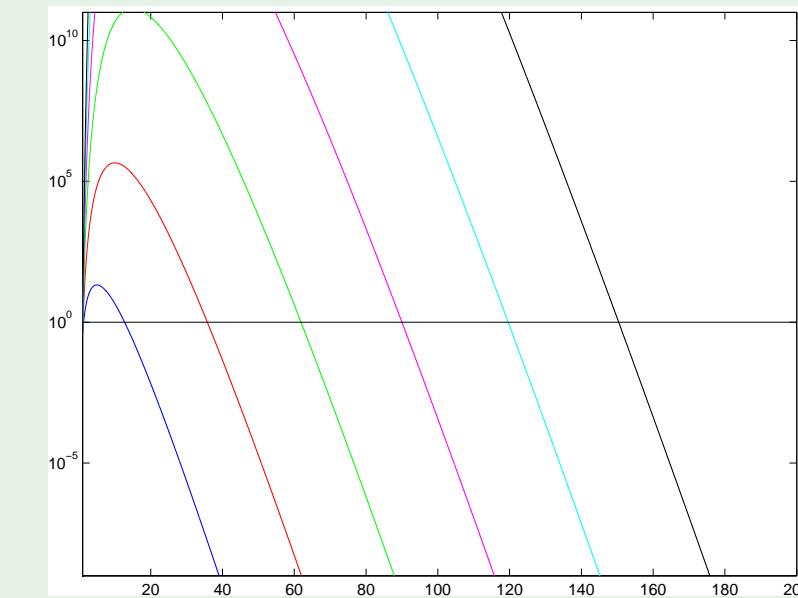
$$E_{\text{out}} \leq E_{\text{in}} + \Omega$$

Review of Lecture 7

- VC dimension $d_{\text{VC}}(\mathcal{H})$
most points \mathcal{H} can shatter
- Scope of VC analysis



- Utility of VC dimension



$$N \propto d_{\text{VC}}$$

Rule of thumb: $N \geq 10 d_{\text{VC}}$

- Generalization bound

$$E_{\text{out}} \leq E_{\text{in}} + \Omega$$

Learning From Data

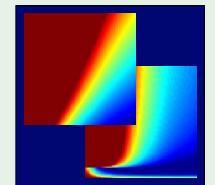
Yaser S. Abu-Mostafa
California Institute of Technology

Lecture 8: Bias-Variance Tradeoff



Sponsored by Caltech's Provost Office, E&AS Division, and IST

• Thursday, April 26, 2012



Outline

- Bias and Variance
- Learning Curves

Approximation-generalization tradeoff

Small E_{out} : good approximation of f out of sample.

More complex $\mathcal{H} \implies$ better chance of **approximating** f

Less complex $\mathcal{H} \implies$ better chance of **generalizing** out of sample

Ideal $\mathcal{H} = \{f\}$ winning lottery ticket ☺

Quantifying the tradeoff

VC analysis was one approach: $E_{\text{out}} \leq E_{\text{in}} + \Omega$

Bias-variance analysis is another: decomposing E_{out} into

1. How well \mathcal{H} can approximate f
2. How well we can zoom in on a good $h \in \mathcal{H}$

Applies to **real-valued targets** and uses **squared error**

Start with E_{out}

$$E_{\text{out}}(g^{(\mathcal{D})}) = \mathbb{E}_{\mathbf{x}} \left[(g^{(\mathcal{D})}(\mathbf{x}) - f(\mathbf{x}))^2 \right]$$

$$\begin{aligned}\mathbb{E}_{\mathcal{D}} [E_{\text{out}}(g^{(\mathcal{D})})] &= \mathbb{E}_{\mathcal{D}} \left[\mathbb{E}_{\mathbf{x}} \left[(g^{(\mathcal{D})}(\mathbf{x}) - f(\mathbf{x}))^2 \right] \right] \\ &= \mathbb{E}_{\mathbf{x}} \left[\mathbb{E}_{\mathcal{D}} \left[(g^{(\mathcal{D})}(\mathbf{x}) - f(\mathbf{x}))^2 \right] \right]\end{aligned}$$

Now, let us focus on:

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

The average hypothesis

To evaluate $\mathbb{E}_{\mathcal{D}} \left[(g^{(\mathcal{D})}(\mathbf{x}) - f(\mathbf{x}))^2 \right]$

we define the ‘average’ hypothesis $\bar{g}(\mathbf{x})$:

$$\bar{g}(\mathbf{x}) = \mathbb{E}_{\mathcal{D}} \left[g^{(\mathcal{D})}(\mathbf{x}) \right]$$

Imagine **many** data sets $\mathcal{D}_1, \mathcal{D}_2, \dots, \mathcal{D}_K$

$$\bar{g}(\mathbf{x}) \approx \frac{1}{K} \sum_{k=1}^K g^{(\mathcal{D}_k)}(\mathbf{x})$$

Using $\bar{g}(\mathbf{x})$

$$\begin{aligned}\mathbb{E}_{\mathcal{D}} \left[(g^{(\mathcal{D})}(\mathbf{x}) - f(\mathbf{x}))^2 \right] &= \mathbb{E}_{\mathcal{D}} \left[(g^{(\mathcal{D})}(\mathbf{x}) - \bar{g}(\mathbf{x}) + \bar{g}(\mathbf{x}) - f(\mathbf{x}))^2 \right] \\ &= \mathbb{E}_{\mathcal{D}} \left[(g^{(\mathcal{D})}(\mathbf{x}) - \bar{g}(\mathbf{x}))^2 + (\bar{g}(\mathbf{x}) - f(\mathbf{x}))^2 \right. \\ &\quad \left. + 2 (g^{(\mathcal{D})}(\mathbf{x}) - \bar{g}(\mathbf{x})) (\bar{g}(\mathbf{x}) - f(\mathbf{x})) \right] \\ &= \mathbb{E}_{\mathcal{D}} \left[(g^{(\mathcal{D})}(\mathbf{x}) - \bar{g}(\mathbf{x}))^2 \right] + (\bar{g}(\mathbf{x}) - f(\mathbf{x}))^2\end{aligned}$$

Bias and variance

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

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

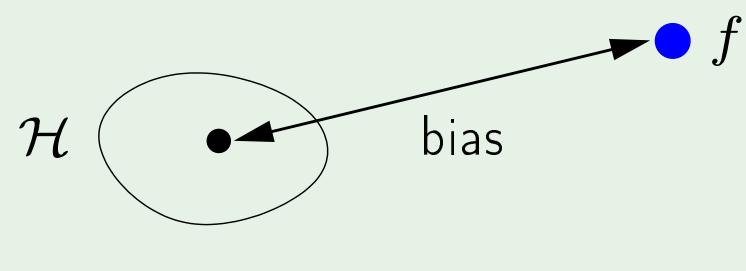
$$= \mathbb{E}_{\mathbf{x}} [\text{bias}(\mathbf{x}) + \text{var}(\mathbf{x})]$$

$$= \text{bias} + \text{var}$$

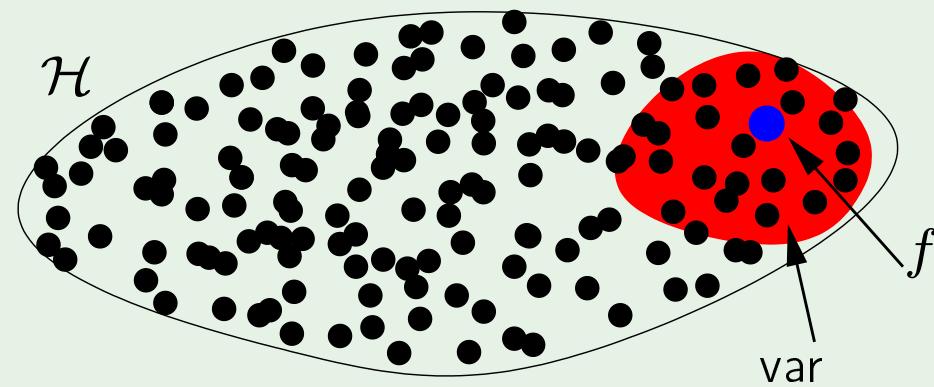
The tradeoff

$$\text{bias} = \mathbb{E}_{\mathbf{x}} \left[(\bar{g}(\mathbf{x}) - f(\mathbf{x}))^2 \right]$$

$$\text{var} = \mathbb{E}_{\mathbf{x}} \left[\mathbb{E}_{\mathcal{D}} \left[(g^{(\mathcal{D})}(\mathbf{x}) - \bar{g}(\mathbf{x}))^2 \right] \right]$$



$\mathcal{H} \uparrow$



Example: sine target

$$f : [-1, 1] \rightarrow \mathbb{R} \quad f(x) = \sin(\pi x)$$

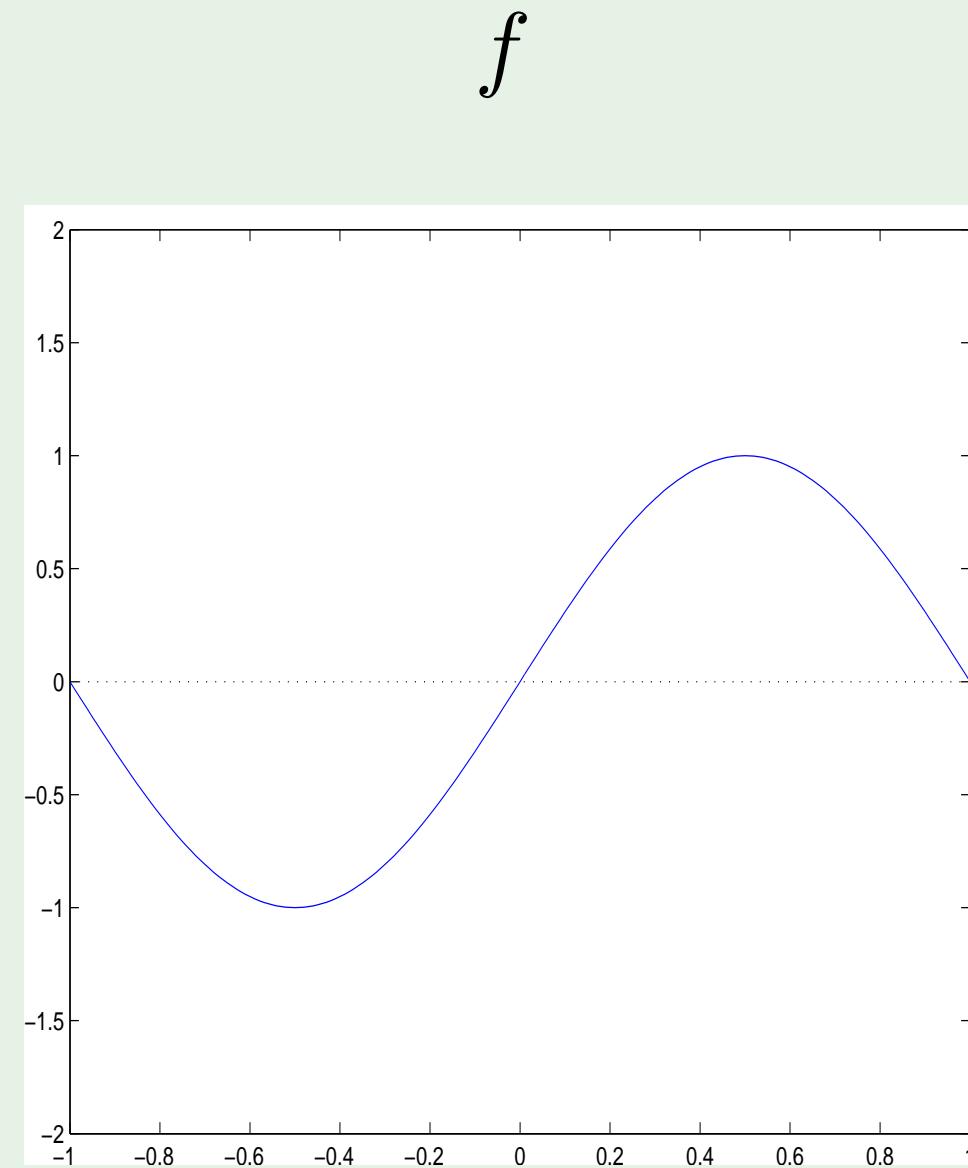
Only two training examples! $N = 2$

Two models used for learning:

$$\mathcal{H}_0: \quad h(x) = b$$

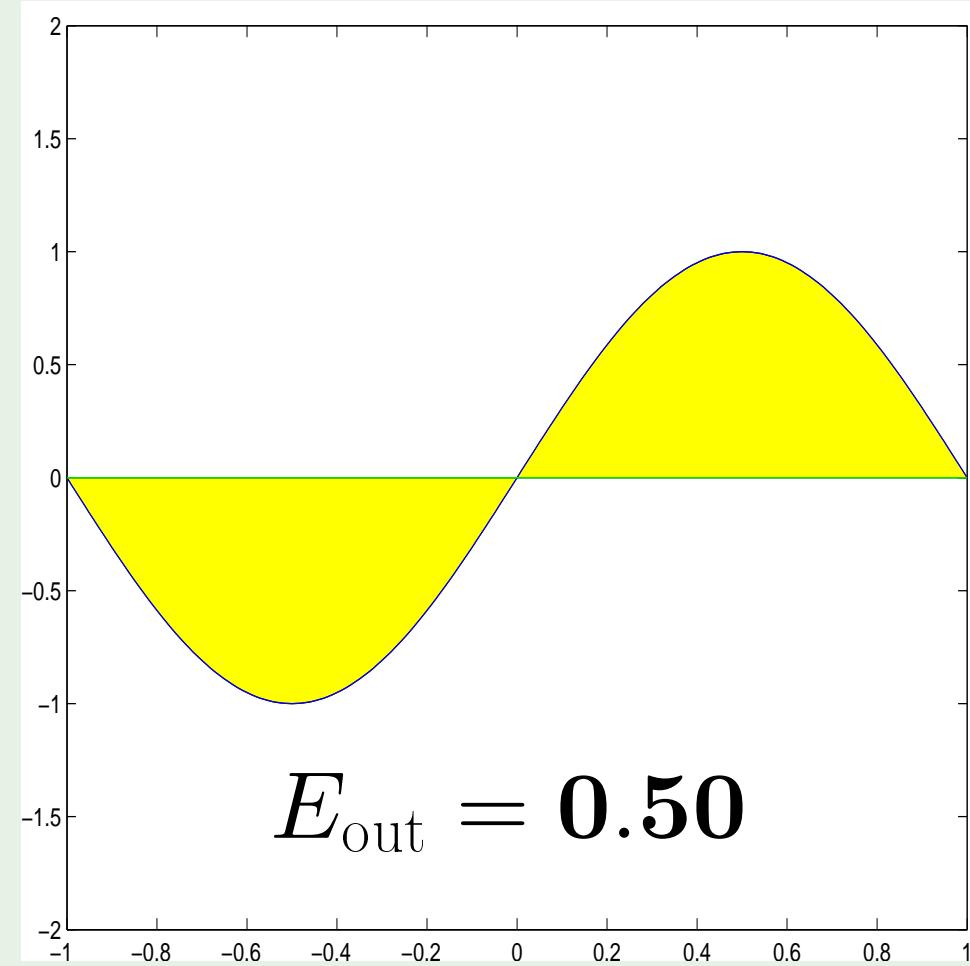
$$\mathcal{H}_1: \quad h(x) = ax + b$$

Which is better, \mathcal{H}_0 or \mathcal{H}_1 ?

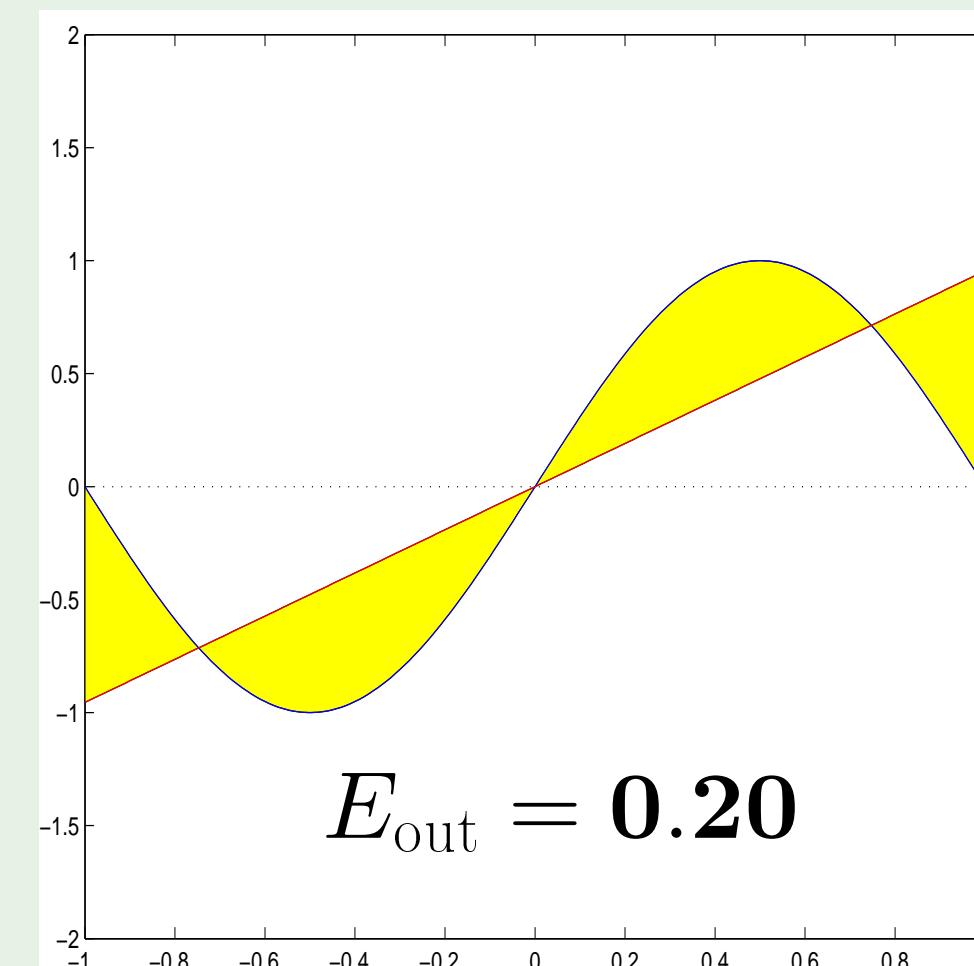


Approximation - \mathcal{H}_0 versus \mathcal{H}_1

\mathcal{H}_0

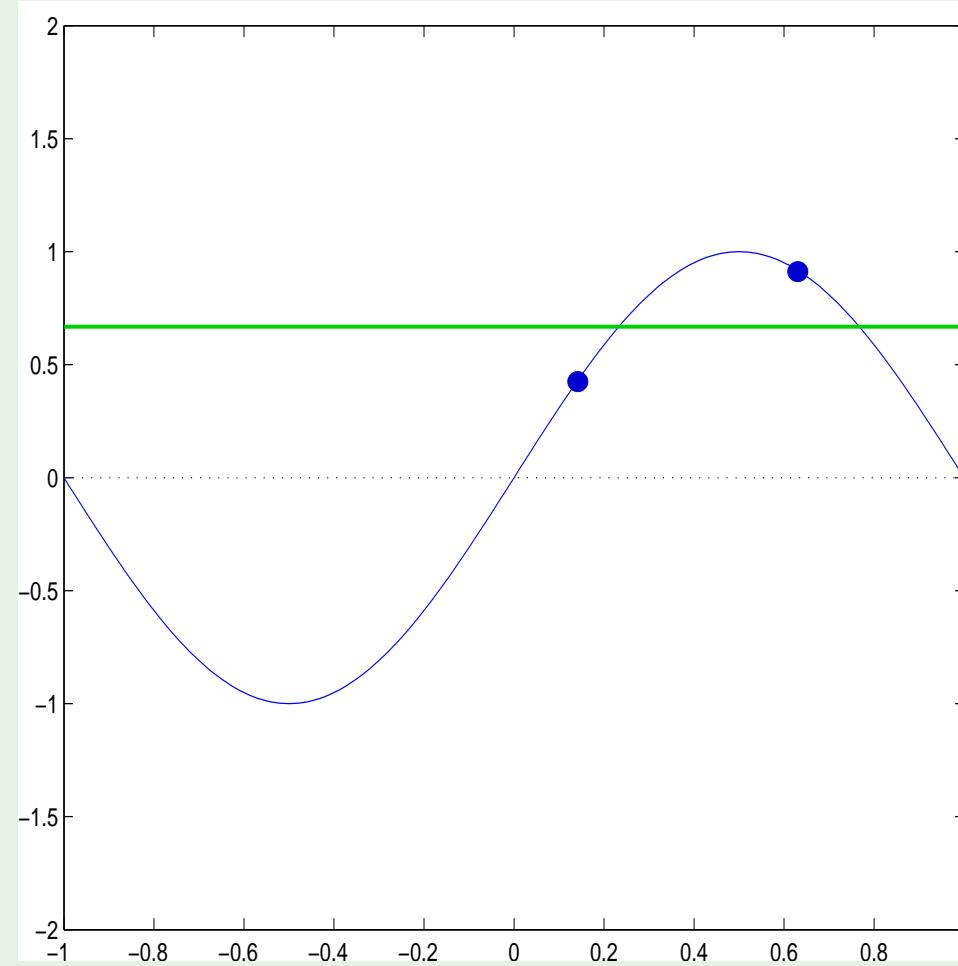


\mathcal{H}_1

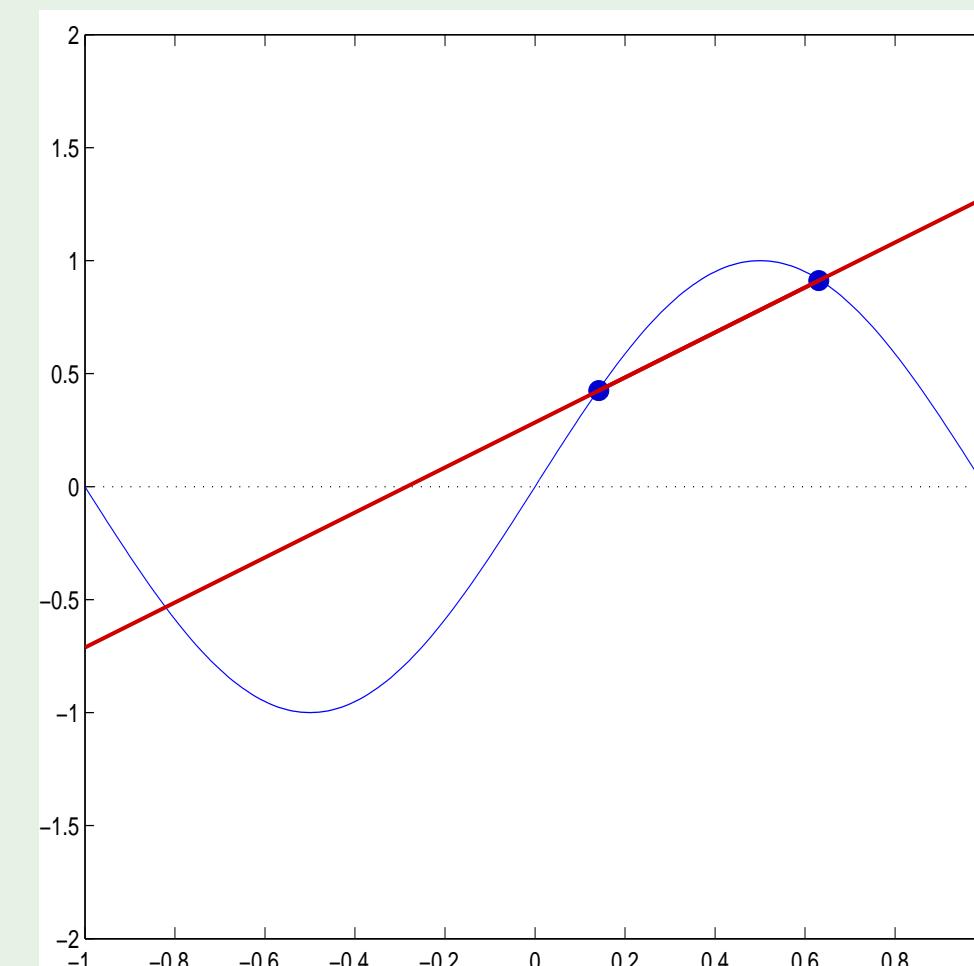


Learning - \mathcal{H}_0 versus \mathcal{H}_1

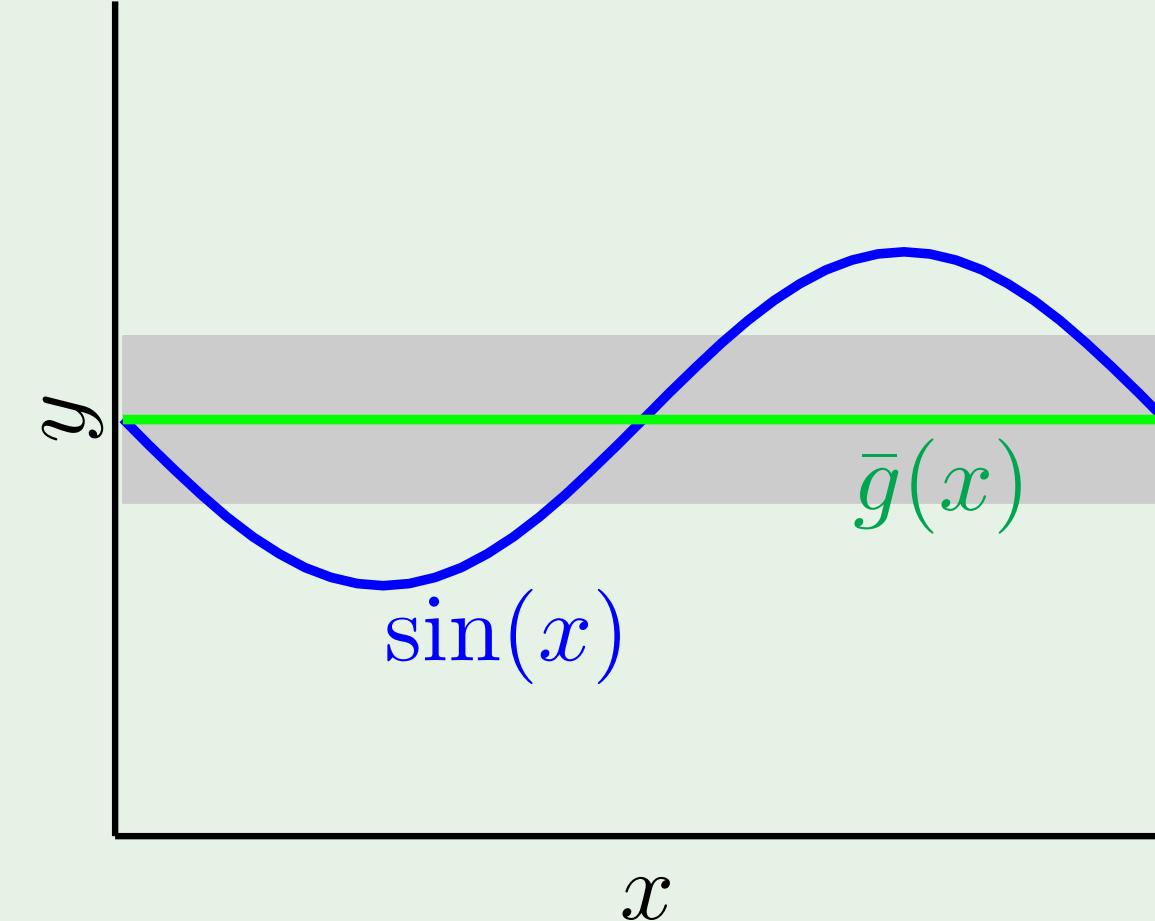
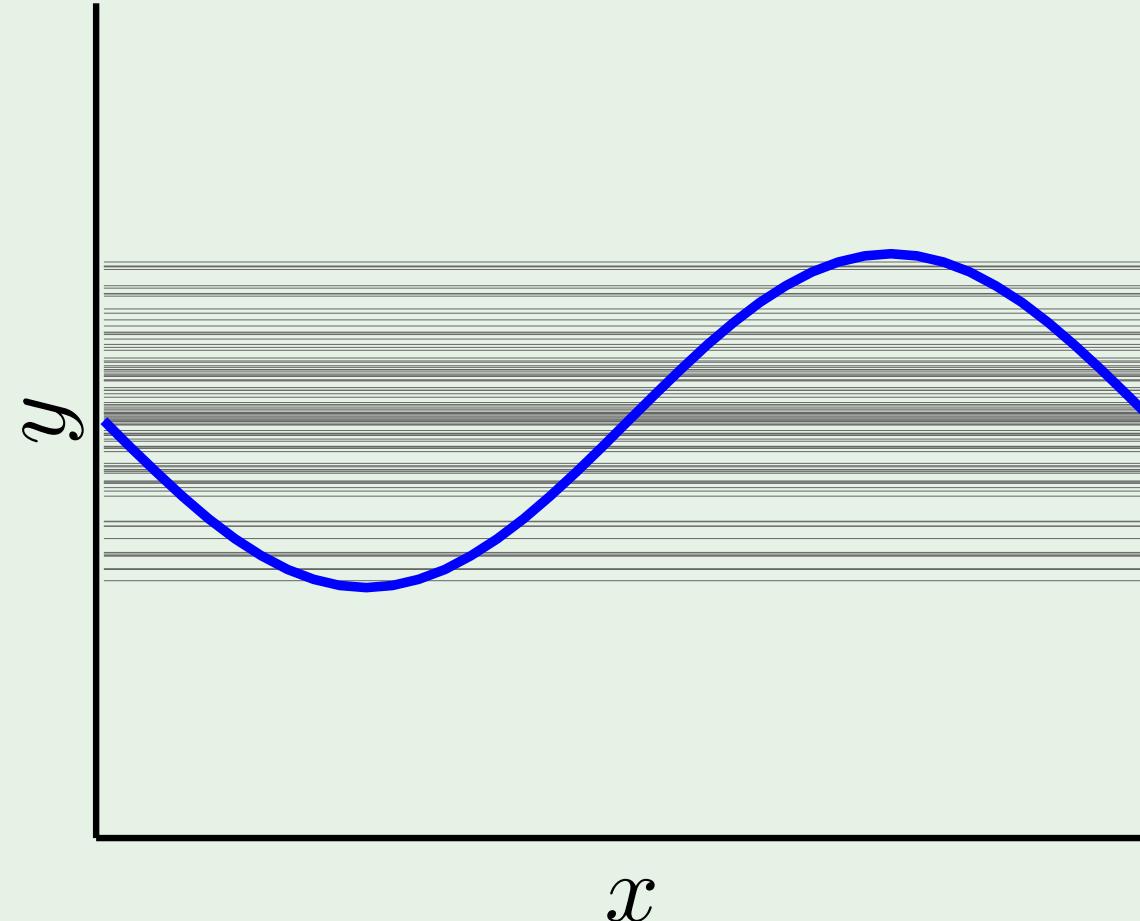
\mathcal{H}_0



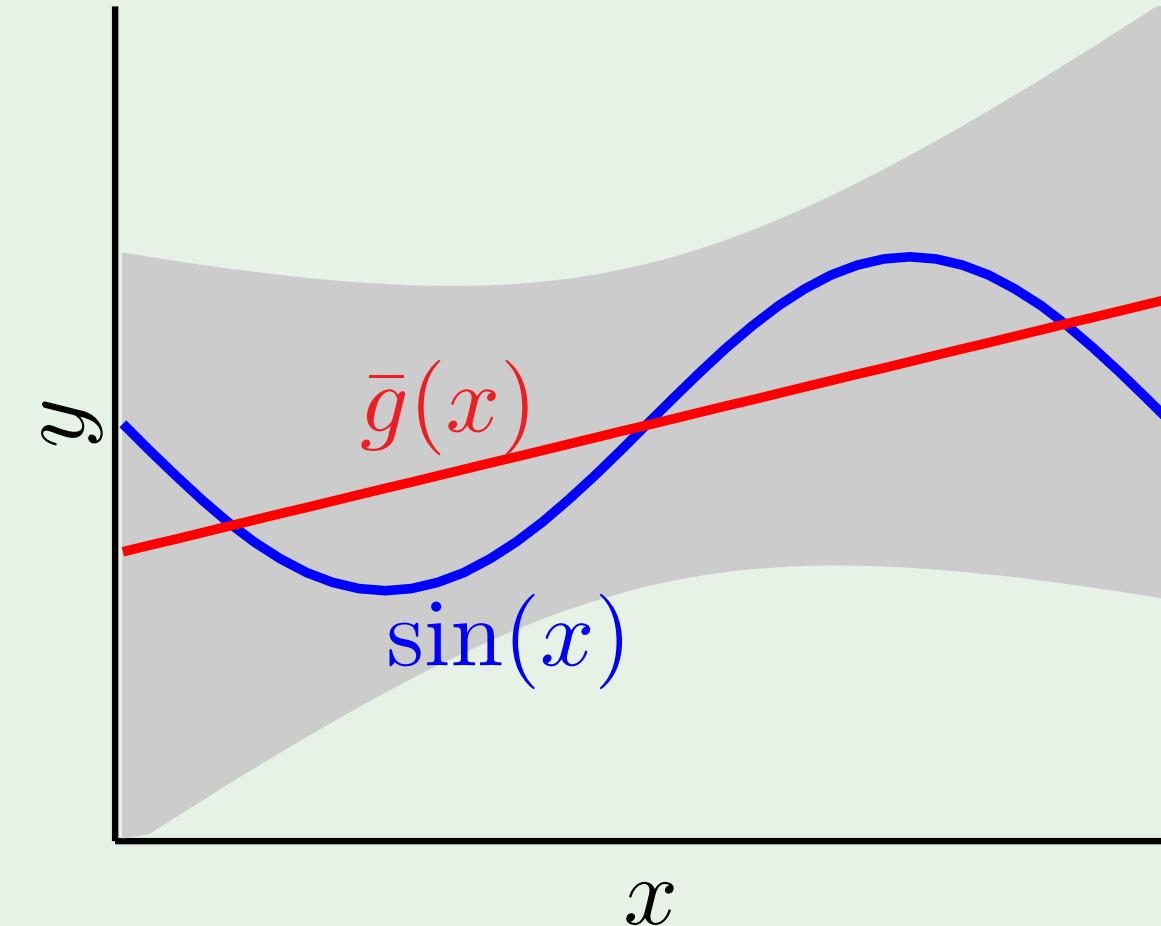
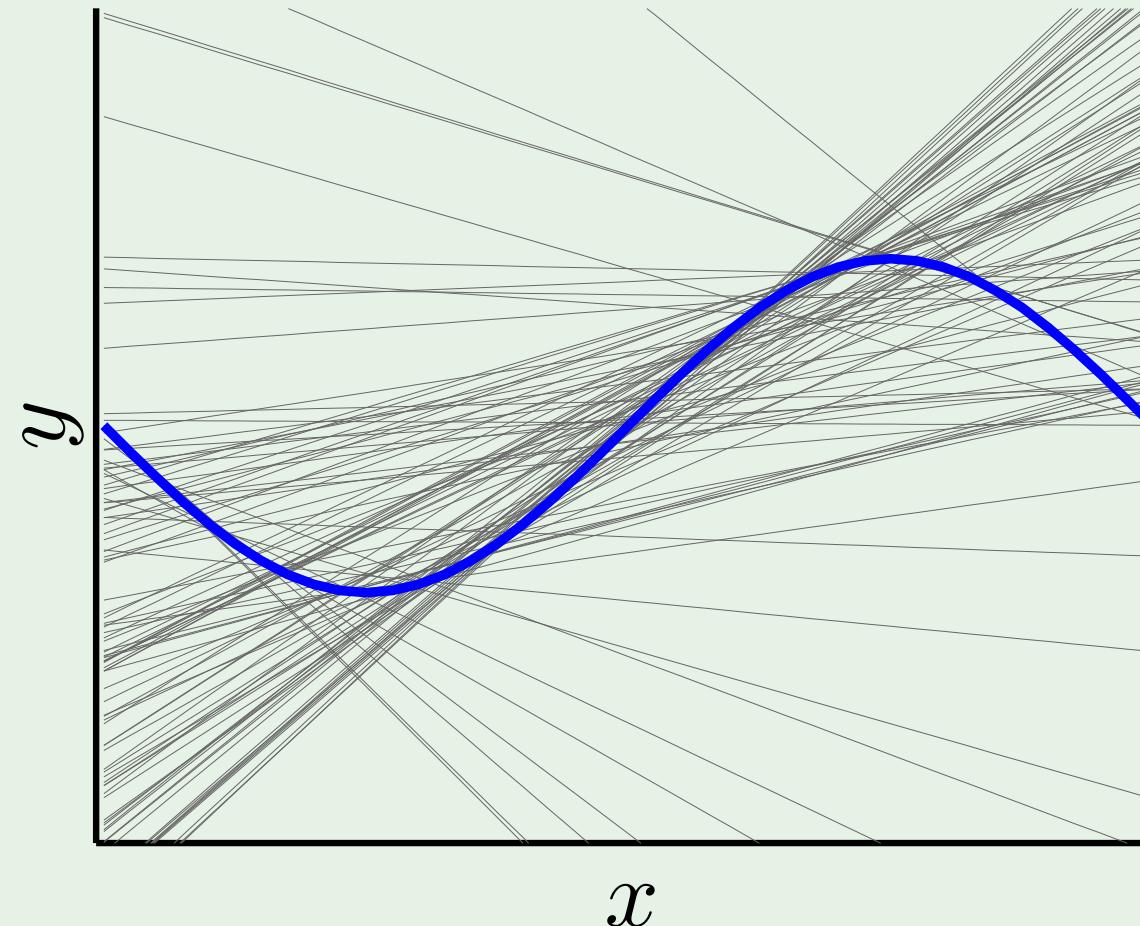
\mathcal{H}_1



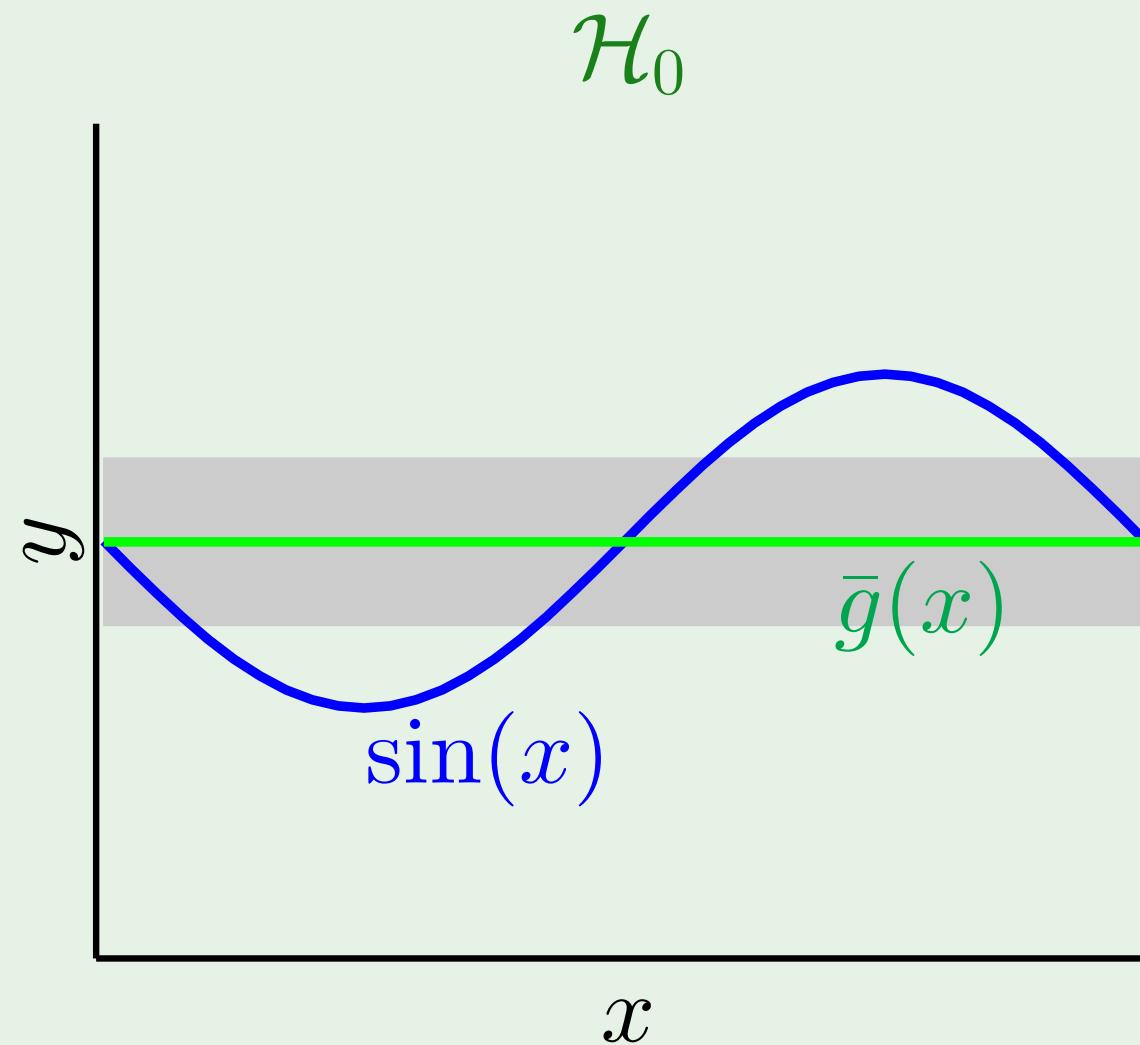
Bias and variance - \mathcal{H}_0



Bias and variance - \mathcal{H}_1

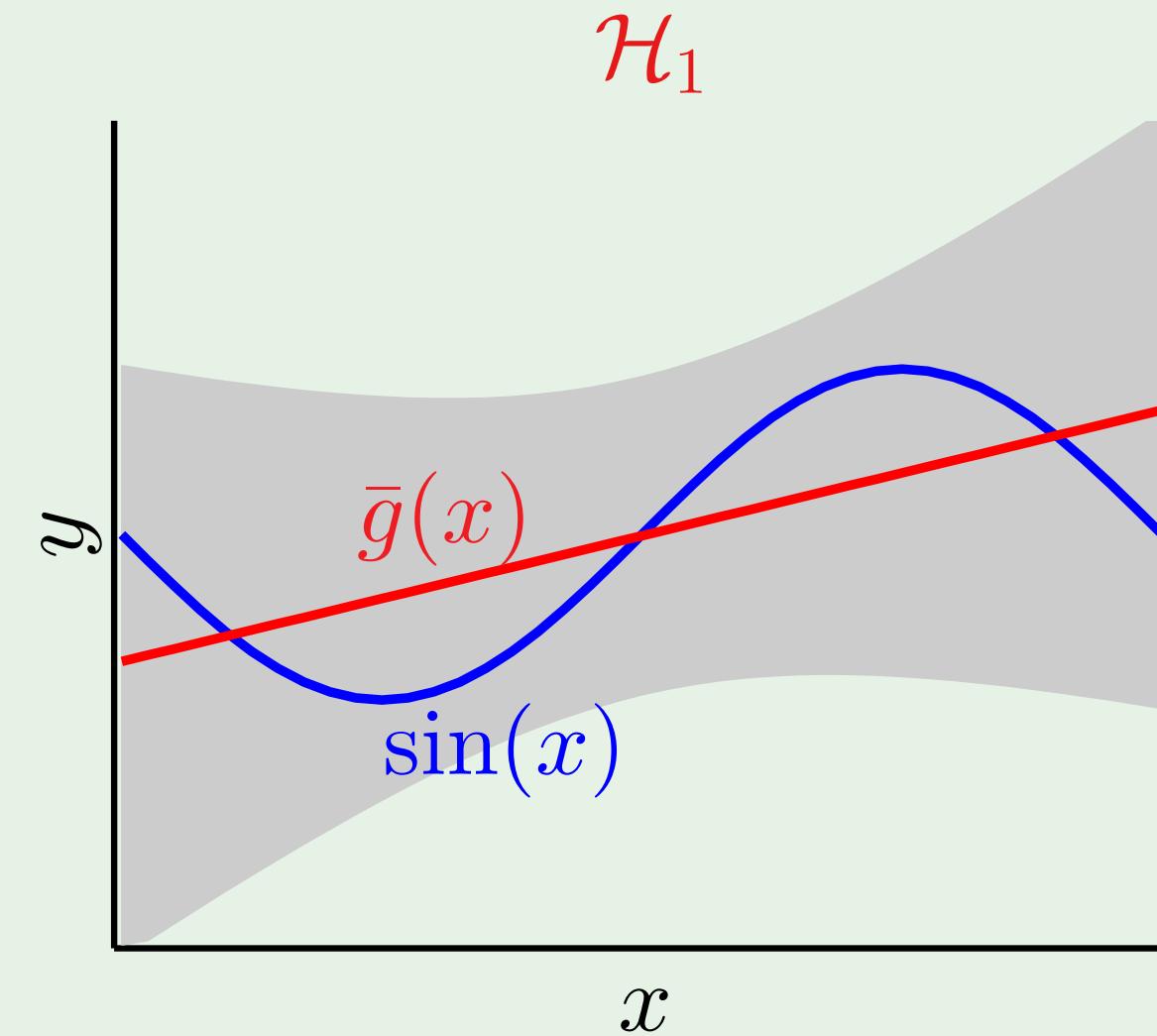


and the winner is ...



bias = **0.50**

var = **0.25**



bias = **0.21**

var = **1.69**

Lesson learned

Match the ‘model complexity’

to the **data resources**, not to the **target complexity**

Outline

- Bias and Variance
- Learning Curves

Expected E_{out} and E_{in}

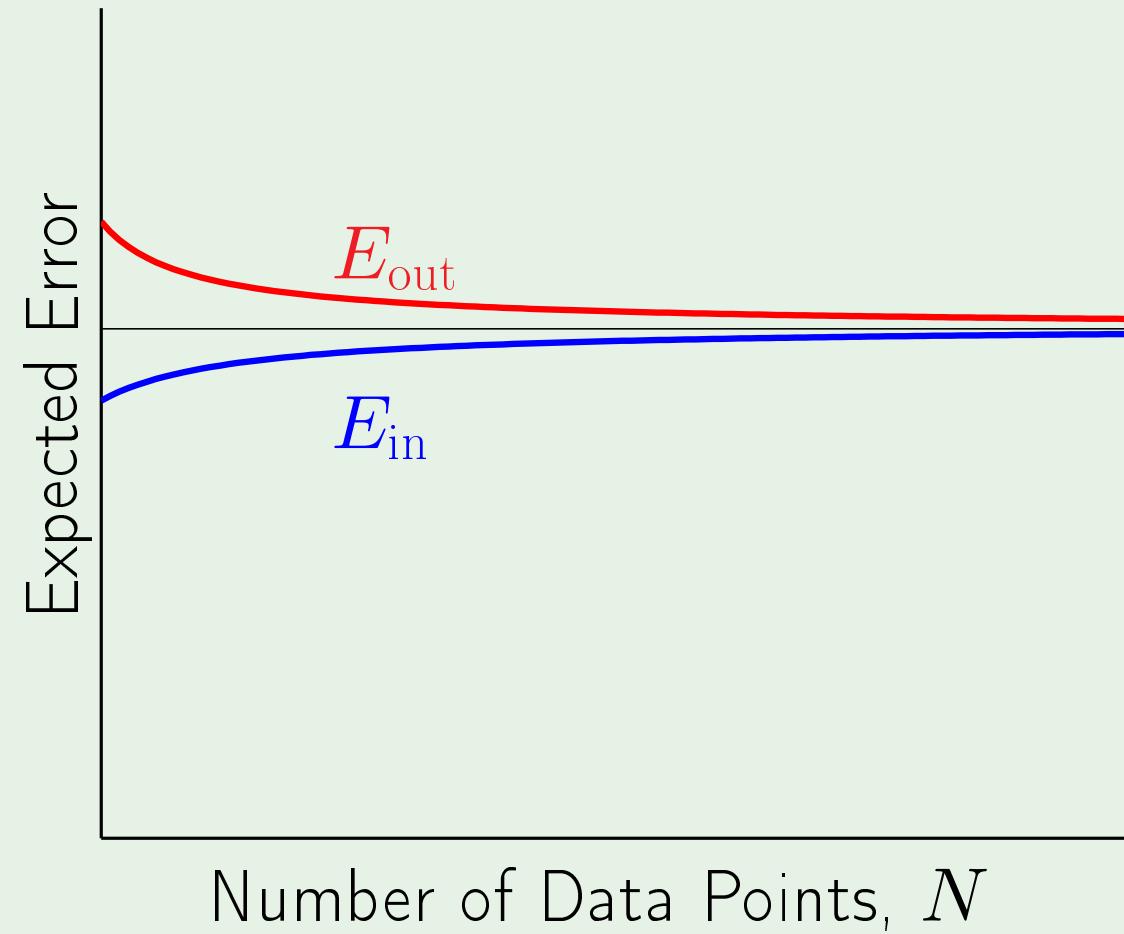
Data set \mathcal{D} of size N

Expected out-of-sample error $\mathbb{E}_{\mathcal{D}}[E_{\text{out}}(g^{(\mathcal{D})})]$

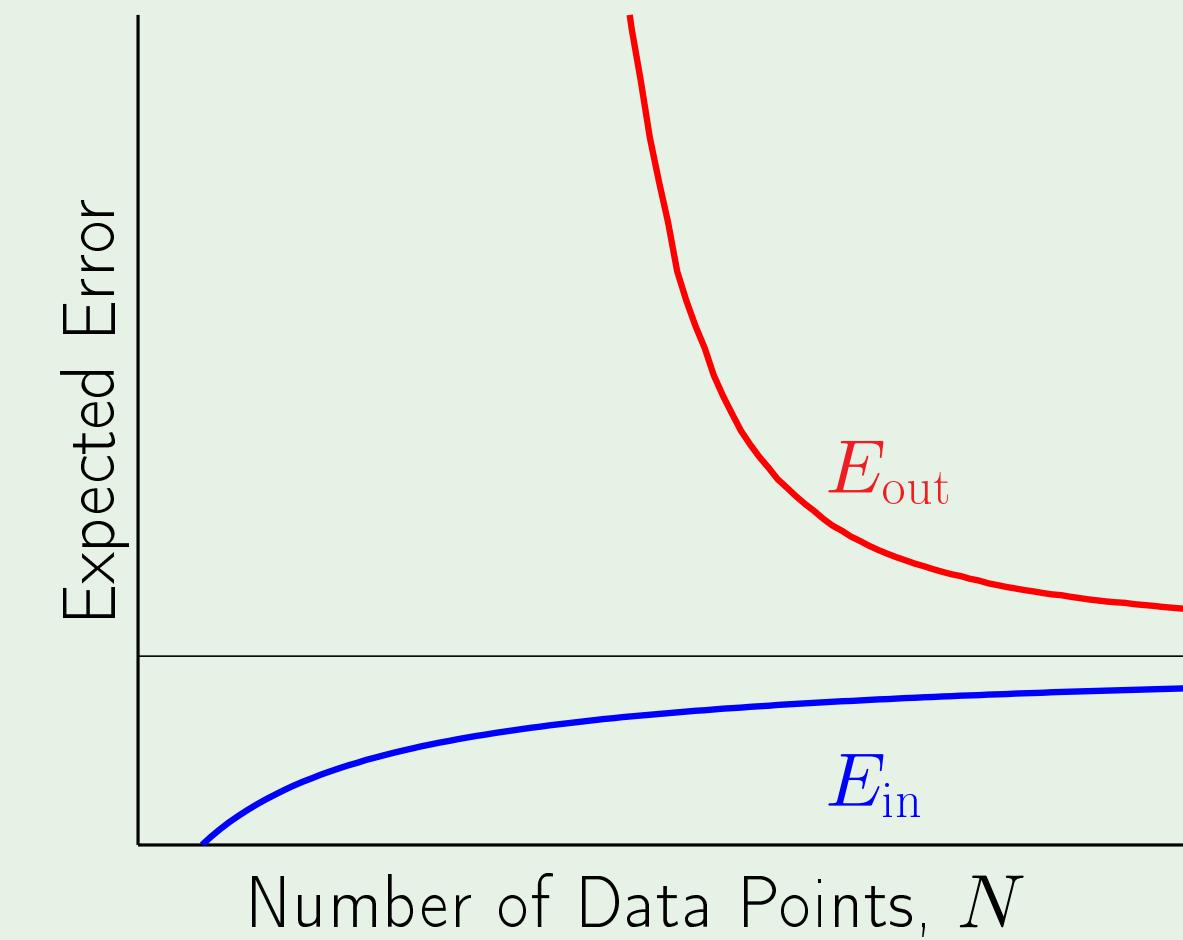
Expected in-sample error $\mathbb{E}_{\mathcal{D}}[E_{\text{in}}(g^{(\mathcal{D})})]$

How do they vary with N ?

The curves

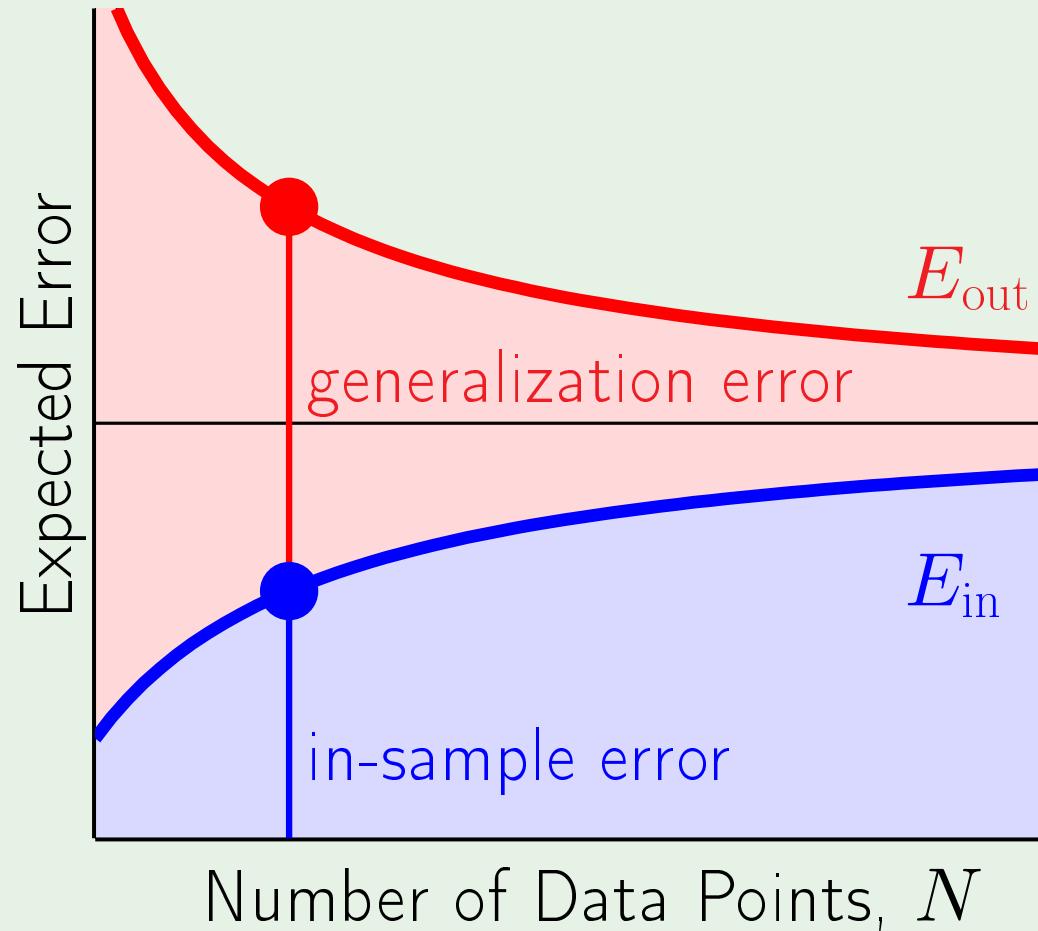


Simple Model

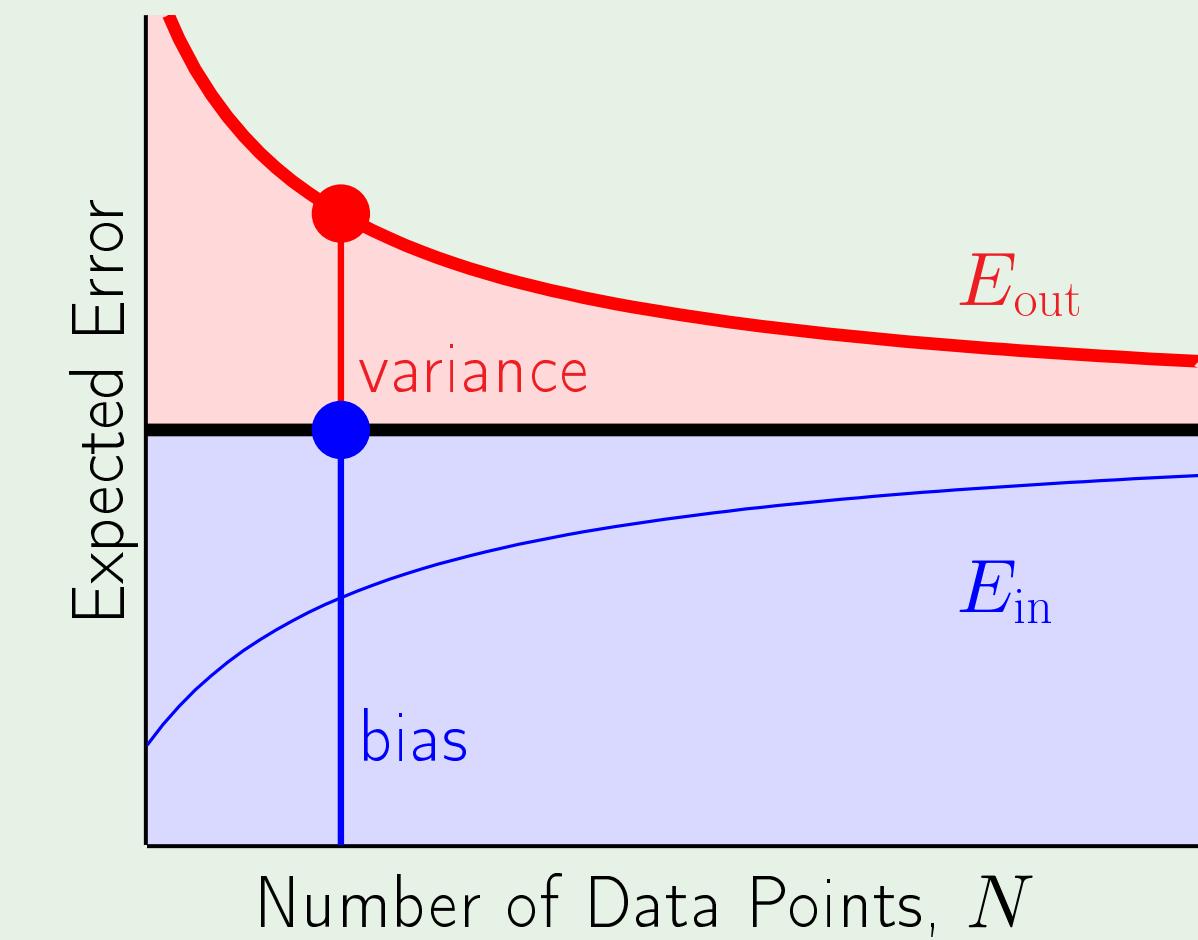


Complex Model

VC versus bias-variance



VC analysis



bias-variance

Linear regression case

Noisy target $y = \mathbf{w}^{*\top} \mathbf{x} + \text{noise}$

Data set $\mathcal{D} = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)\}$

Linear regression solution: $\mathbf{w} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}$

In-sample error vector = $\mathbf{X}\mathbf{w} - \mathbf{y}$

'Out-of-sample' error vector = $\mathbf{X}\mathbf{w} - \mathbf{y}'$

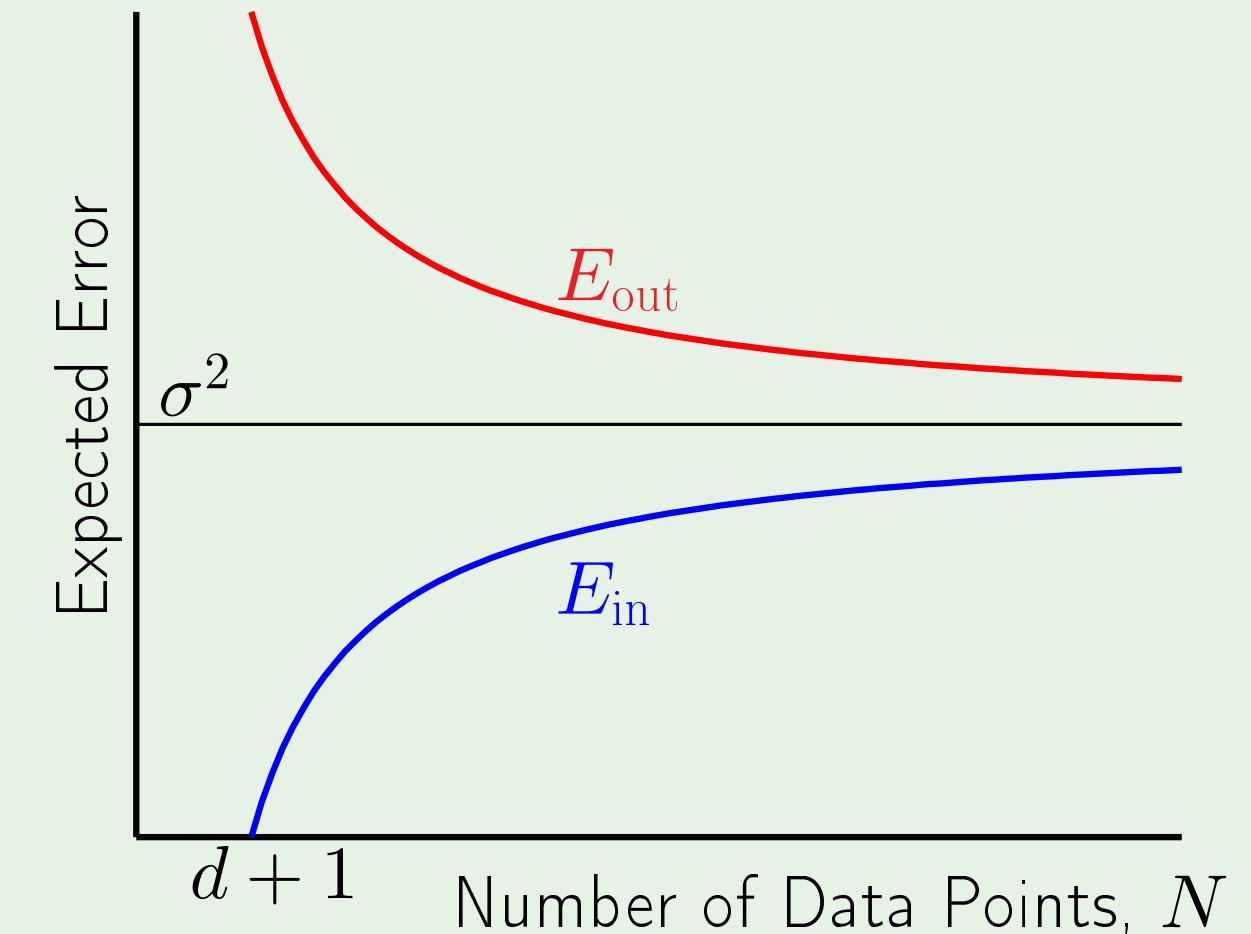
Learning curves for linear regression

Best approximation error = σ^2

Expected in-sample error = $\sigma^2 \left(1 - \frac{d+1}{N}\right)$

Expected out-of-sample error = $\sigma^2 \left(1 + \frac{d+1}{N}\right)$

Expected generalization error = $2\sigma^2 \left(\frac{d+1}{N}\right)$

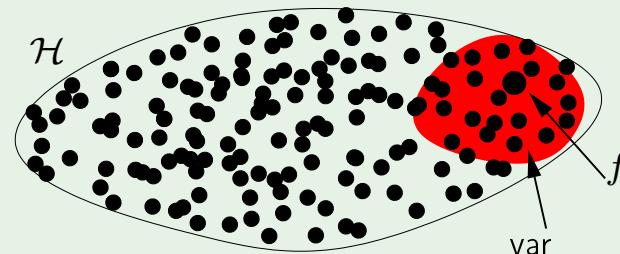
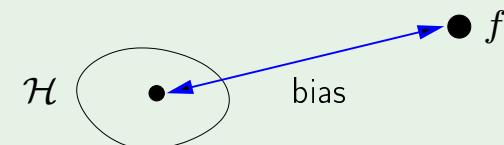


Review of Lecture 8

- Bias and variance

Expected value of E_{out} w.r.t. \mathcal{D}

$$= \text{bias} + \text{var}$$

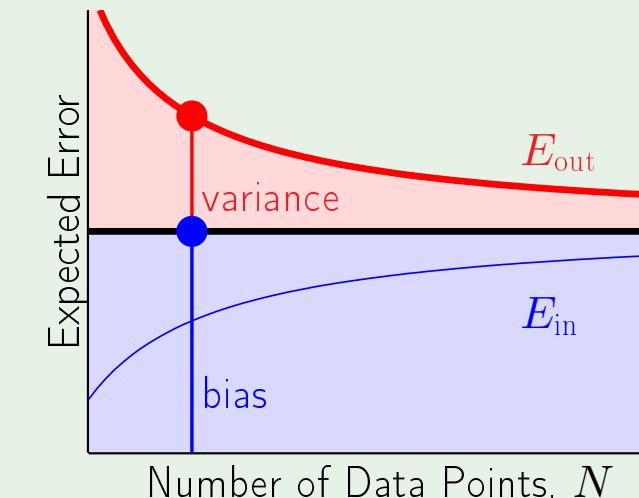


$$g^{(\mathcal{D})}(\mathbf{x}) \rightarrow \bar{g}(\mathbf{x}) \rightarrow f(\mathbf{x})$$

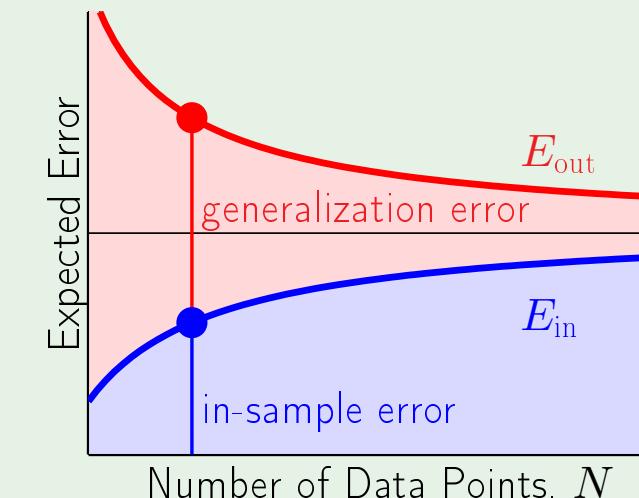
- Learning curves

How E_{in} and E_{out} vary with N

B-V:



VC:



- $N \propto \text{"VC dimension"}$

Learning From Data

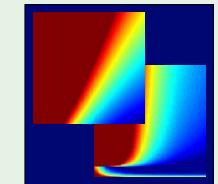
Yaser S. Abu-Mostafa
California Institute of Technology

Lecture 9: The Linear Model II



Sponsored by Caltech's Provost Office, E&AS Division, and IST

• Tuesday, May 1, 2012



Where we are

- Linear classification ✓
- Linear regression ✓
- Logistic regression
- Nonlinear transforms ✎

Nonlinear transforms

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

Each $z_i = \phi_i(\mathbf{x})$ $\mathbf{z} = \Phi(\mathbf{x})$

Example: $\mathbf{z} = (1, x_1, x_2, x_1x_2, x_1^2, x_2^2)$

Final hypothesis $g(\mathbf{x})$ in \mathcal{X} space:

$$\text{sign} (\tilde{\mathbf{w}}^\top \Phi(\mathbf{x})) \quad \text{or} \quad \tilde{\mathbf{w}}^\top \Phi(\mathbf{x})$$

The price we pay

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



\mathbf{w}

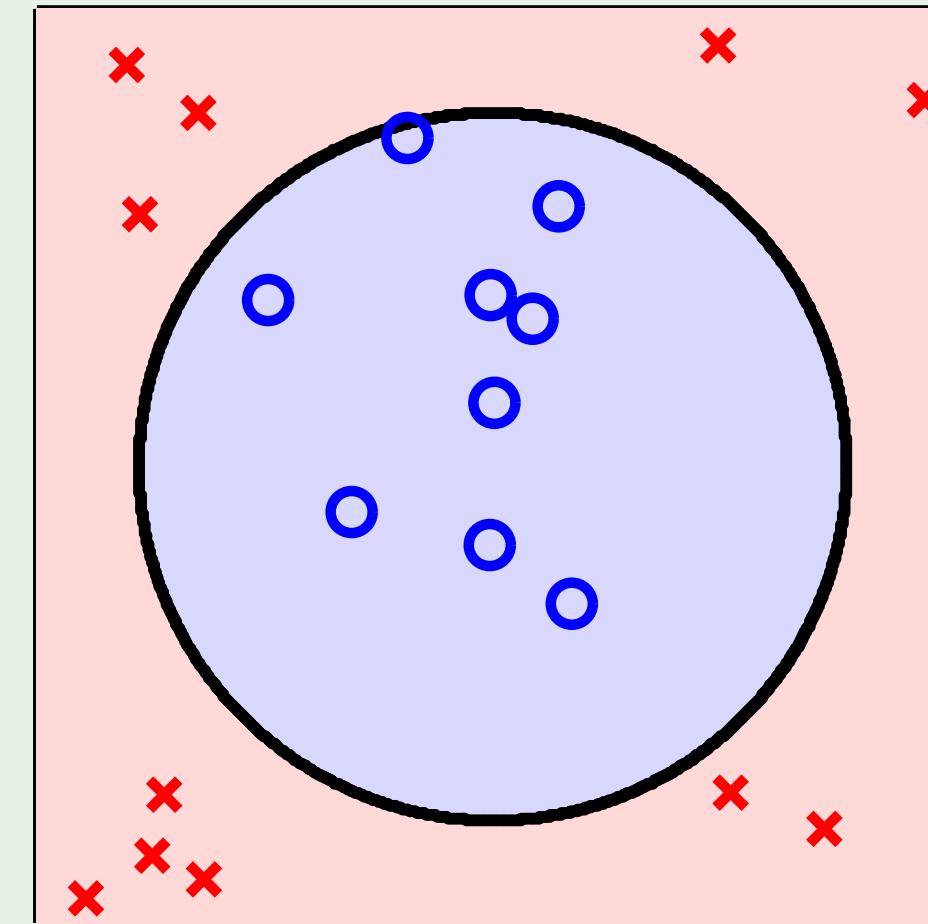
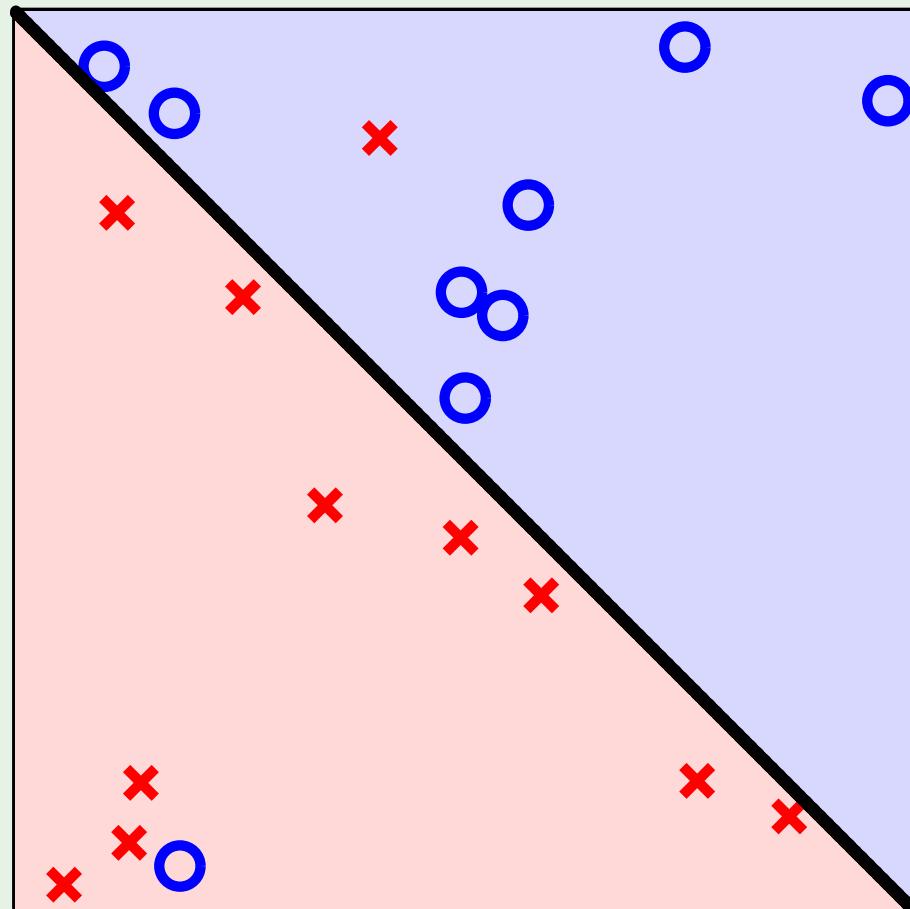


$\tilde{\mathbf{w}}$

$$d_{\text{VC}} = d + 1$$

$$d_{\text{VC}} \leq \tilde{d} + 1$$

Two non-separable cases

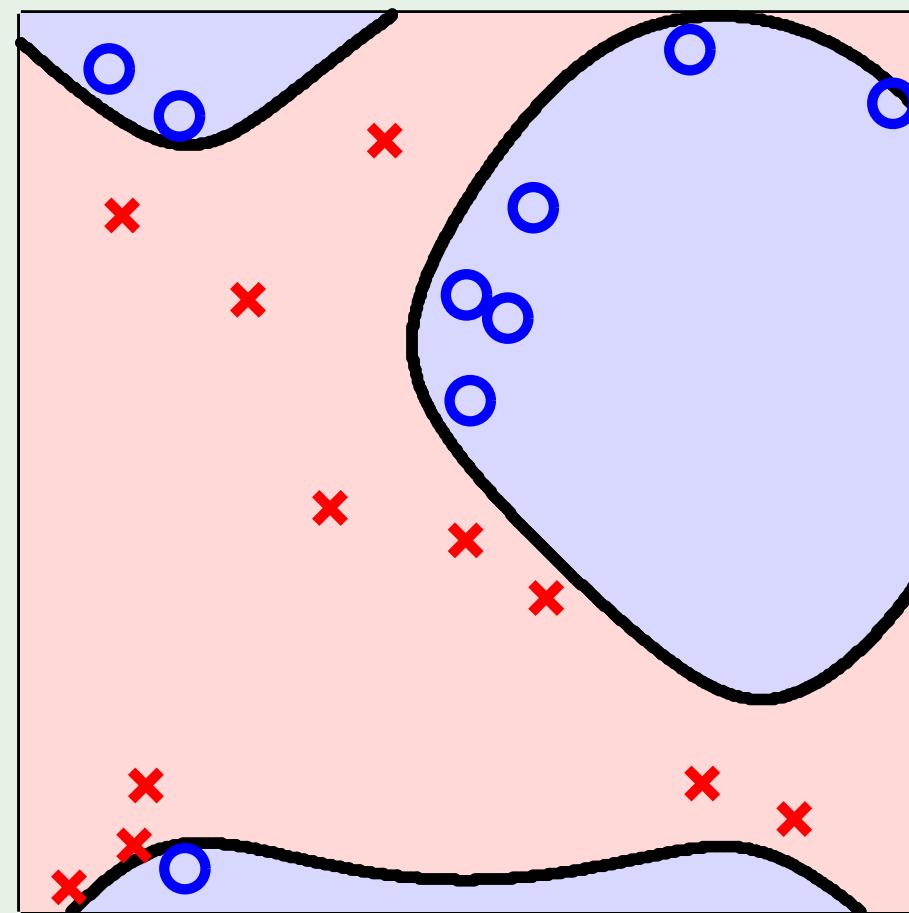


First case

Use a linear model in \mathcal{X} ; accept $E_{\text{in}} > 0$

or

Insist on $E_{\text{in}} = 0$; go to high-dimensional \mathcal{Z}



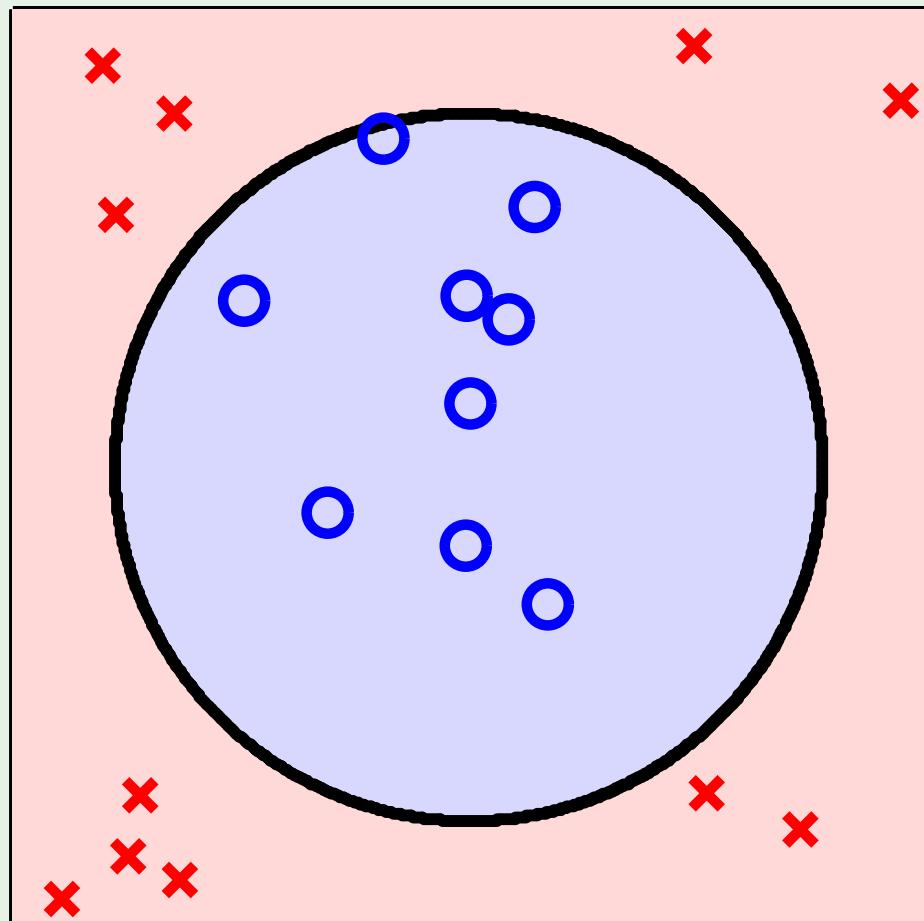
Second case

$$\mathbf{z} = (1, x_1, x_2, x_1x_2, x_1^2, x_2^2)$$

Why not: $\mathbf{z} = (1, x_1^2, x_2^2)$

or better yet: $\mathbf{z} = (1, x_1^2 + x_2^2)$

or even: $\mathbf{z} = (x_1^2 + x_2^2 - 0.6)$



Lesson learned

Looking at the data *before* choosing the model can be hazardous to your E_{out}

Data snooping



Logistic regression - Outline

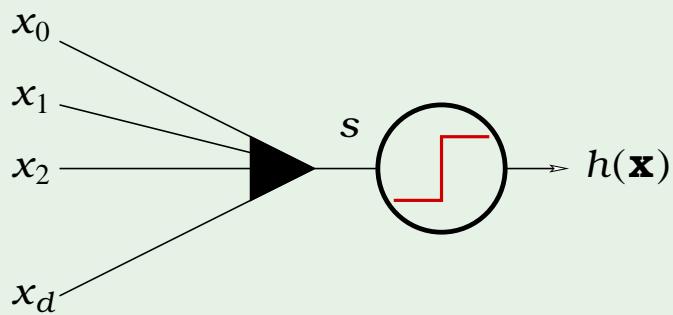
- The model
- Error measure
- Learning algorithm

A third linear model

$$s = \sum_{i=0}^d w_i x_i$$

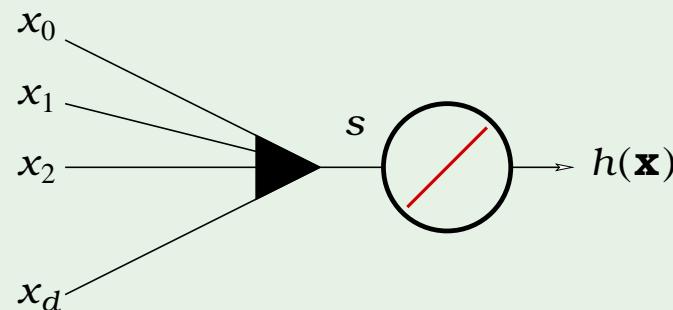
linear classification

$$h(\mathbf{x}) = \text{sign}(s)$$



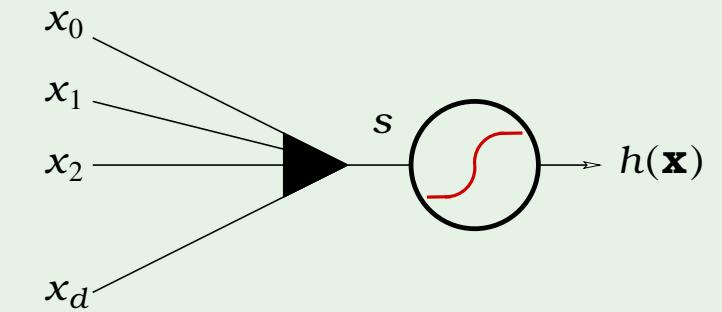
linear regression

$$h(\mathbf{x}) = s$$



logistic regression

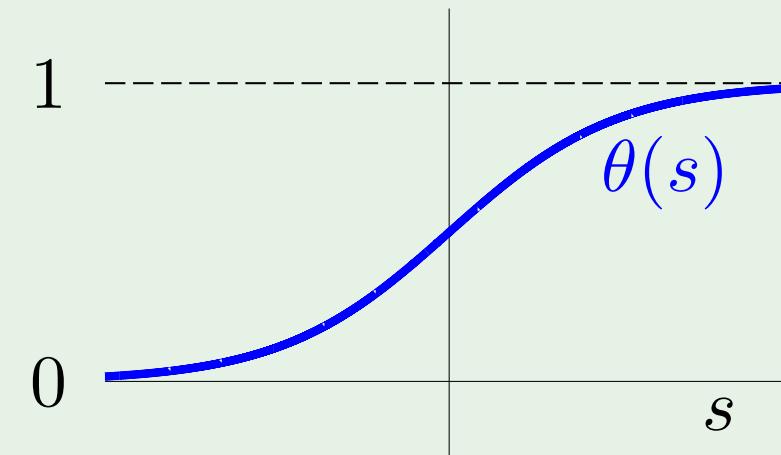
$$h(\mathbf{x}) = \theta(s)$$



The logistic function θ

The formula:

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



soft threshold: uncertainty

sigmoid: flattened out 's'

Probability interpretation

$h(\mathbf{x}) = \theta(s)$ is interpreted as a probability

Example. Prediction of heart attacks

Input \mathbf{x} : cholesterol level, age, weight, etc.

$\theta(s)$: probability of a heart attack

The signal $s = \mathbf{w}^\top \mathbf{x}$ "risk score"

Genuine probability

Data (\mathbf{x}, y) with **binary y** , generated by a noisy target:

$$P(y \mid \mathbf{x}) = \begin{cases} f(\mathbf{x}) & \text{for } y = +1; \\ 1 - f(\mathbf{x}) & \text{for } y = -1. \end{cases}$$

The target $f : \mathbb{R}^d \rightarrow [0, 1]$ is the probability

$$\text{Learn } g(\mathbf{x}) = \theta(\mathbf{w}^\top \mathbf{x}) \approx f(\mathbf{x})$$

Error measure

For each (\mathbf{x}, y) , y is generated by probability $f(\mathbf{x})$

Plausible error measure based on **likelihood**:

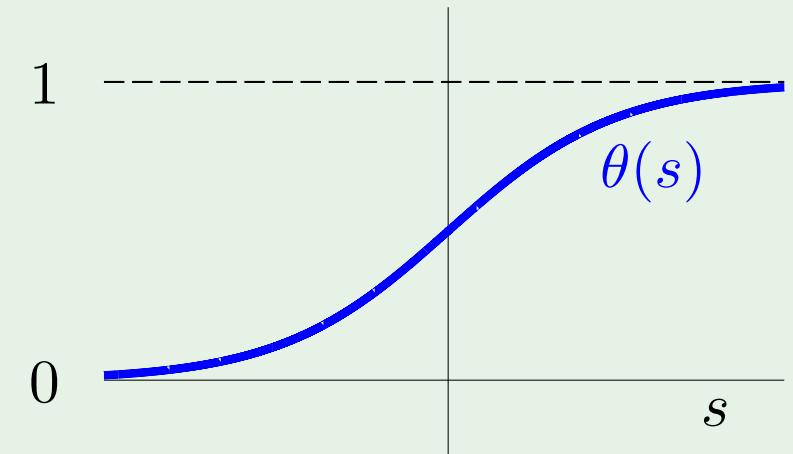
If $h = f$, how likely to get y from \mathbf{x} ?

$$P(y \mid \mathbf{x}) = \begin{cases} h(\mathbf{x}) & \text{for } y = +1; \\ 1 - h(\mathbf{x}) & \text{for } y = -1. \end{cases}$$

Formula for likelihood

$$P(y \mid \mathbf{x}) = \begin{cases} h(\mathbf{x}) & \text{for } y = +1; \\ 1 - h(\mathbf{x}) & \text{for } y = -1. \end{cases}$$

Substitute $h(\mathbf{x}) = \theta(\mathbf{w}^\top \mathbf{x})$, noting $\theta(-s) = 1 - \theta(s)$



$$P(y \mid \mathbf{x}) = \theta(y \mathbf{w}^\top \mathbf{x})$$

Likelihood of $\mathcal{D} = (\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)$ is

$$\prod_{n=1}^N P(y_n \mid \mathbf{x}_n) = \prod_{n=1}^N \theta(y_n \mathbf{w}^\top \mathbf{x}_n)$$

Maximizing the likelihood

Minimize

$$-\frac{1}{N} \ln \left(\prod_{n=1}^N \theta(y_n \mathbf{w}^\top \mathbf{x}_n) \right)$$

$$= \frac{1}{N} \sum_{n=1}^N \ln \left(\frac{1}{\theta(y_n \mathbf{w}^\top \mathbf{x}_n)} \right)$$

$$\left[\theta(s) = \frac{1}{1 + e^{-s}} \right]$$

$$E_{\text{in}}(\mathbf{w}) = \frac{1}{N} \sum_{n=1}^N \underbrace{\ln \left(1 + e^{-y_n \mathbf{w}^\top \mathbf{x}_n} \right)}_{\text{e}(h(\mathbf{x}_n), y_n)}$$

“cross-entropy” error

Logistic regression - Outline

- The model
- Error measure
- Learning algorithm

How to minimize E_{in}

For logistic regression,

$$E_{\text{in}}(\mathbf{w}) = \frac{1}{N} \sum_{n=1}^N \ln \left(1 + e^{-y_n \mathbf{w}^\top \mathbf{x}_n} \right) \quad \longleftarrow \text{iterative solution}$$

Compare to linear regression:

$$E_{\text{in}}(\mathbf{w}) = \frac{1}{N} \sum_{n=1}^N (\mathbf{w}^\top \mathbf{x}_n - y_n)^2 \quad \longleftarrow \text{closed-form solution}$$

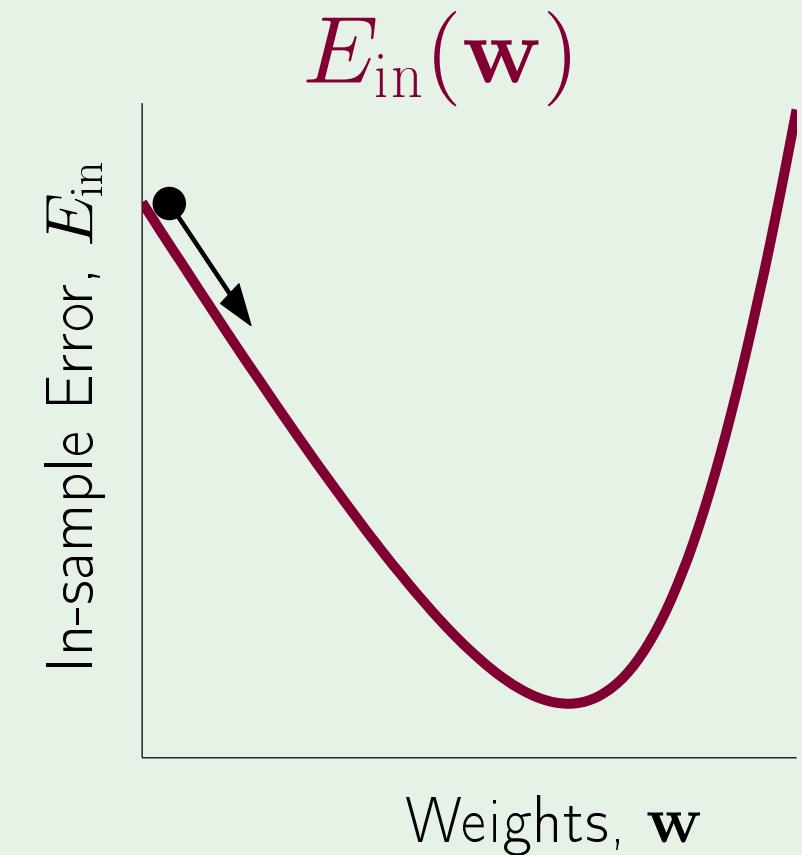
Iterative method: gradient descent

General method for nonlinear optimization

Start at $\mathbf{w}(0)$; take a step along steepest slope

Fixed step size: $\mathbf{w}(1) = \mathbf{w}(0) + \eta \hat{\mathbf{v}}$

What is the direction $\hat{\mathbf{v}}$?



Formula for the direction $\hat{\mathbf{v}}$

$$\Delta E_{\text{in}} = E_{\text{in}}(\mathbf{w}(0) + \eta \hat{\mathbf{v}}) - E_{\text{in}}(\mathbf{w}(0))$$

$$= \eta \nabla E_{\text{in}}(\mathbf{w}(0))^T \hat{\mathbf{v}} + O(\eta^2)$$

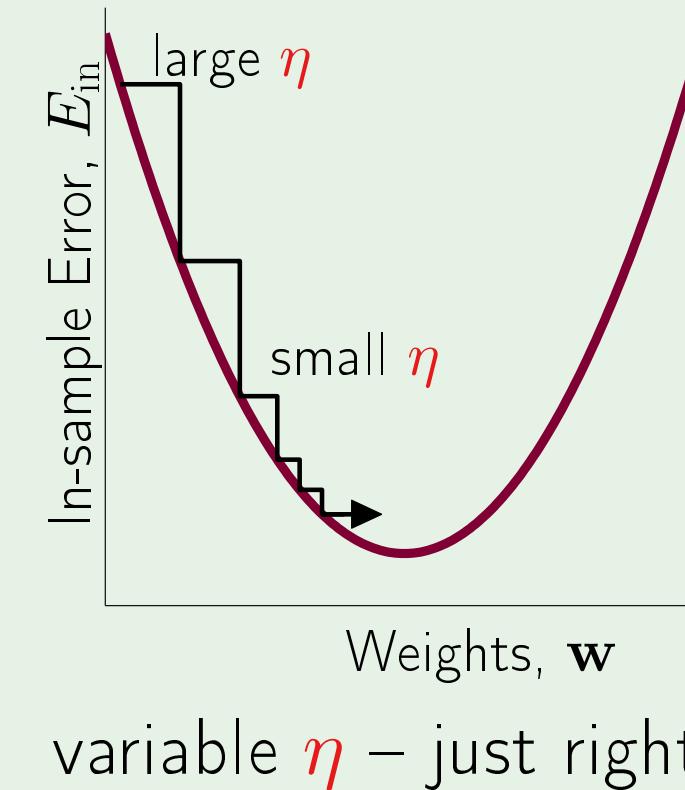
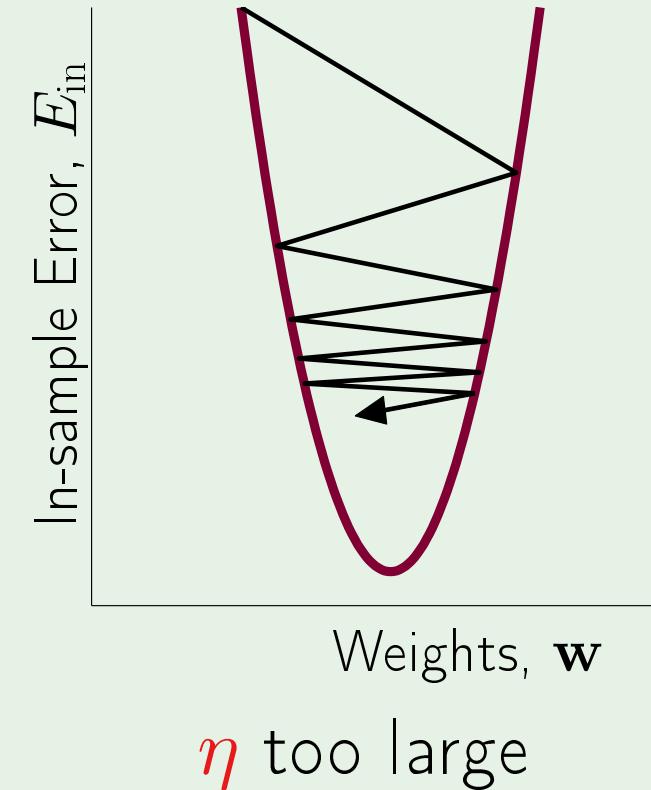
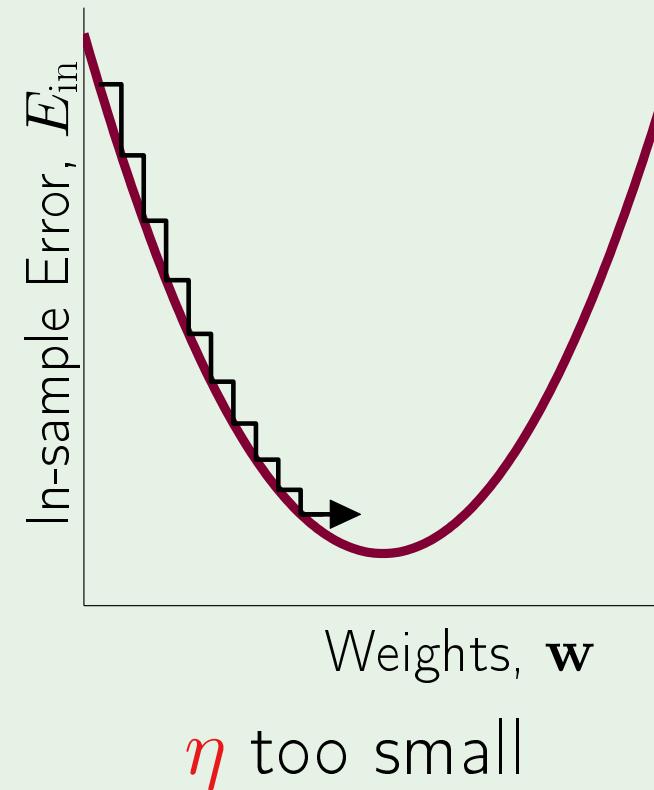
$$\geq -\eta \|\nabla E_{\text{in}}(\mathbf{w}(0))\|$$

Since $\hat{\mathbf{v}}$ is a unit vector,

$$\hat{\mathbf{v}} = - \frac{\nabla E_{\text{in}}(\mathbf{w}(0))}{\|\nabla E_{\text{in}}(\mathbf{w}(0))\|}$$

Fixed-size step?

How η affects the algorithm:



η should increase with the slope

Easy implementation

Instead of

$$\Delta \mathbf{w} = \eta \hat{\mathbf{v}}$$

$$= -\eta \frac{\nabla E_{\text{in}}(\mathbf{w}(0))}{\|\nabla E_{\text{in}}(\mathbf{w}(0))\|}$$

Have

$$\Delta \mathbf{w} = -\eta \nabla E_{\text{in}}(\mathbf{w}(0))$$

Fixed learning rate η

Logistic regression algorithm

1: Initialize the weights at $t = 0$ to $\mathbf{w}(0)$

2: **for** $t = 0, 1, 2, \dots$ **do**

3: Compute the gradient

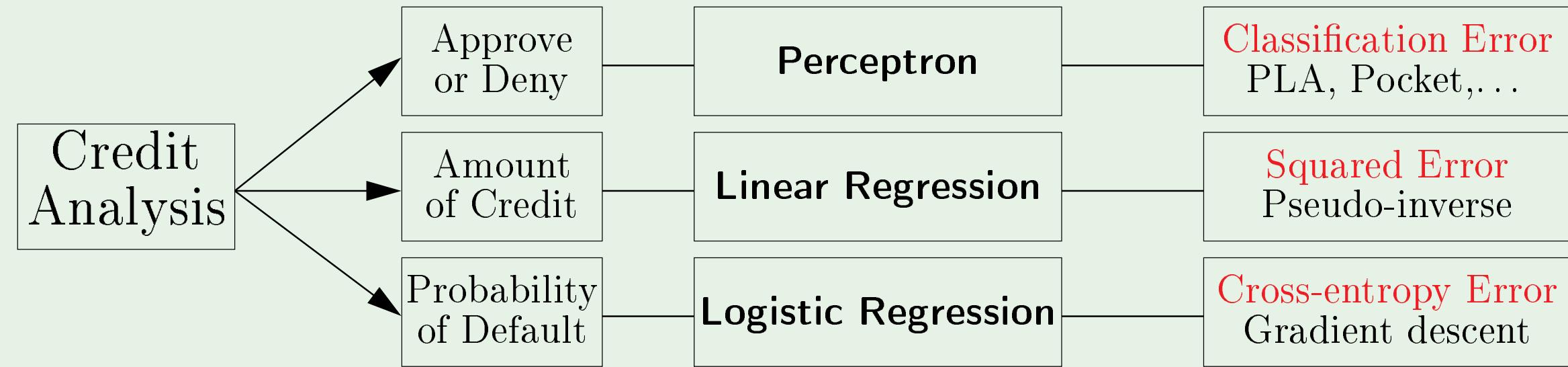
$$\nabla E_{\text{in}} = -\frac{1}{N} \sum_{n=1}^N \frac{y_n \mathbf{x}_n}{1 + e^{y_n \mathbf{w}^\top(t) \mathbf{x}_n}}$$

4: Update the weights: $\mathbf{w}(t+1) = \mathbf{w}(t) - \eta \nabla E_{\text{in}}$

5: Iterate to the next step until it is time to stop

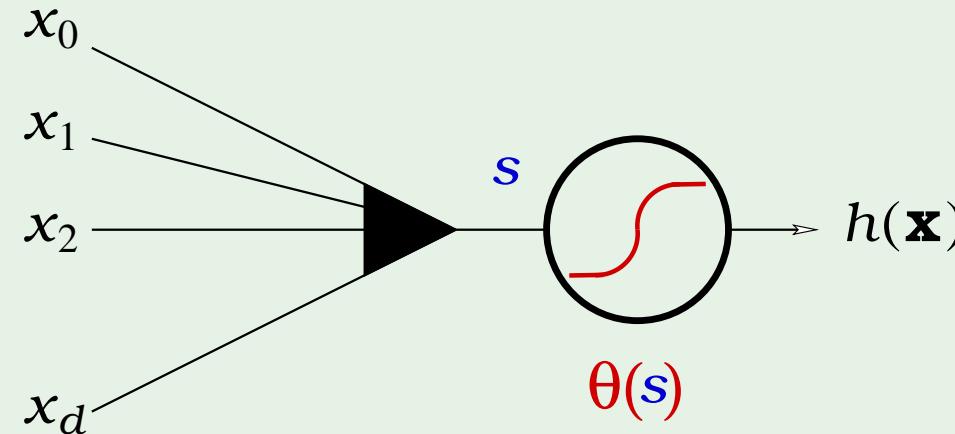
6: Return the final weights \mathbf{w}

Summary of Linear Models



Review of Lecture 9

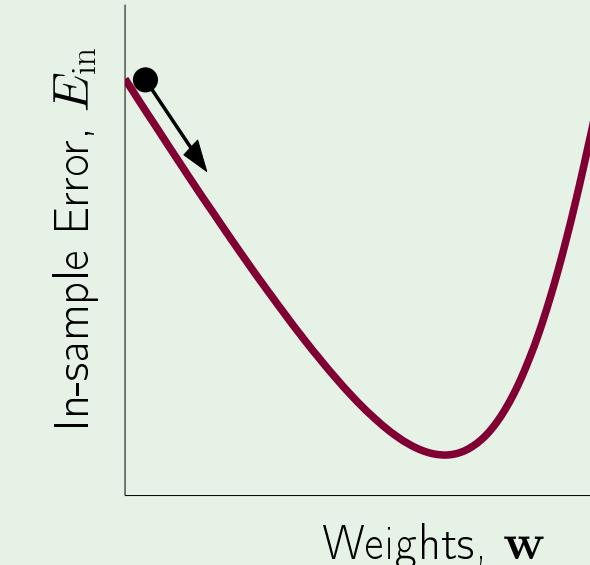
- Logistic regression



- Likelihood measure

$$\prod_{n=1}^N P(y_n \mid \mathbf{x}_n) = \prod_{n=1}^N \theta(y_n \mathbf{w}^\top \mathbf{x}_n)$$

- Gradient descent



- Initialize $\mathbf{w}(0)$
- For $t = 0, 1, 2, \dots$ [to termination]
 - $\mathbf{w}(t + 1) = \mathbf{w}(t) - \eta \nabla E_{\text{in}}(\mathbf{w}(t))$
- Return final \mathbf{w}

Learning From Data

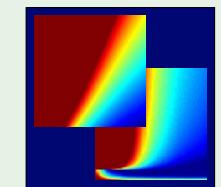
Yaser S. Abu-Mostafa
California Institute of Technology

Lecture 10: Neural Networks



Sponsored by Caltech's Provost Office, E&AS Division, and IST

• Thursday, May 3, 2012



Outline

- Stochastic gradient descent
- Neural network model
- Backpropagation algorithm

Stochastic gradient descent

GD minimizes:

$$E_{\text{in}}(\mathbf{w}) = \frac{1}{N} \sum_{n=1}^N \underbrace{\mathbf{e}(\mathbf{h}(\mathbf{x}_n), y_n)}_{\ln(1+e^{-y_n \mathbf{w}^\top \mathbf{x}_n})} \quad \leftarrow \text{in logistic regression}$$

by iterative steps along $-\nabla E_{\text{in}}$:

$$\Delta \mathbf{w} = -\eta \nabla E_{\text{in}}(\mathbf{w})$$

∇E_{in} is based on all examples (\mathbf{x}_n, y_n)

“batch” GD

The stochastic aspect

Pick one (\mathbf{x}_n, y_n) at a time. Apply GD to $\mathbf{e}(h(\mathbf{x}_n), y_n)$

“Average” direction:

$$\begin{aligned}\mathbb{E}_{\mathbf{n}} [-\nabla \mathbf{e}(h(\mathbf{x}_n), y_n)] &= \frac{1}{N} \sum_{n=1}^N -\nabla \mathbf{e}(h(\mathbf{x}_n), y_n) \\ &= -\nabla E_{\text{in}}\end{aligned}$$

randomized version of GD

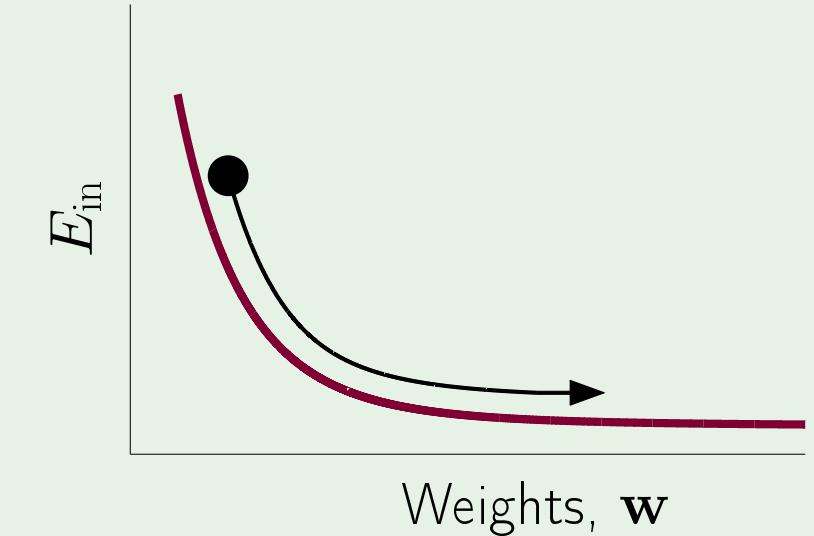
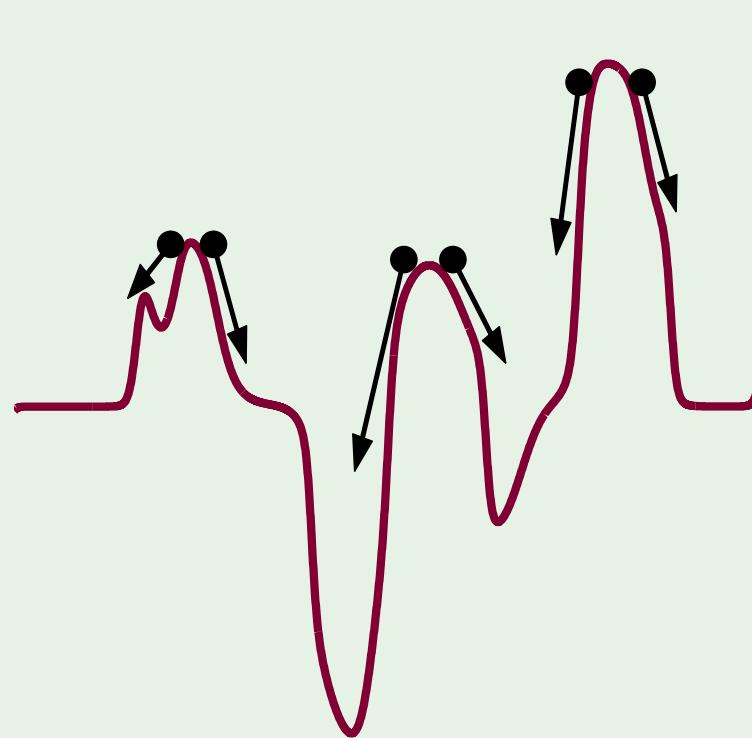
stochastic gradient descent (SGD)

Benefits of SGD

1. cheaper computation
2. randomization
3. simple

Rule of thumb:

$\eta = 0.1$ works

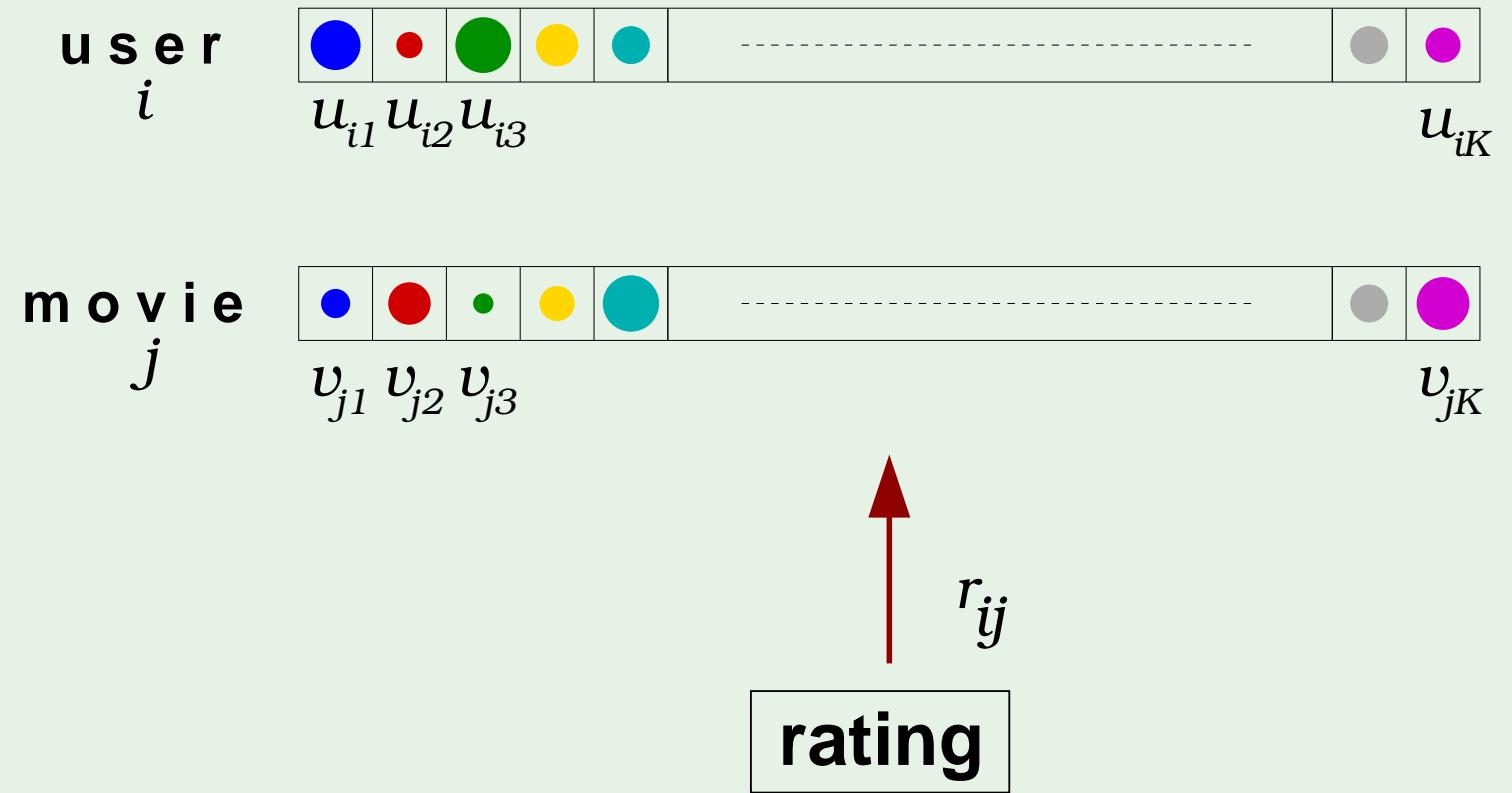


randomization helps

SGD in action

Remember movie ratings?

$$e_{ij} = \left(r_{ij} - \sum_{k=1}^K u_{ik} v_{jk} \right)^2$$



Outline

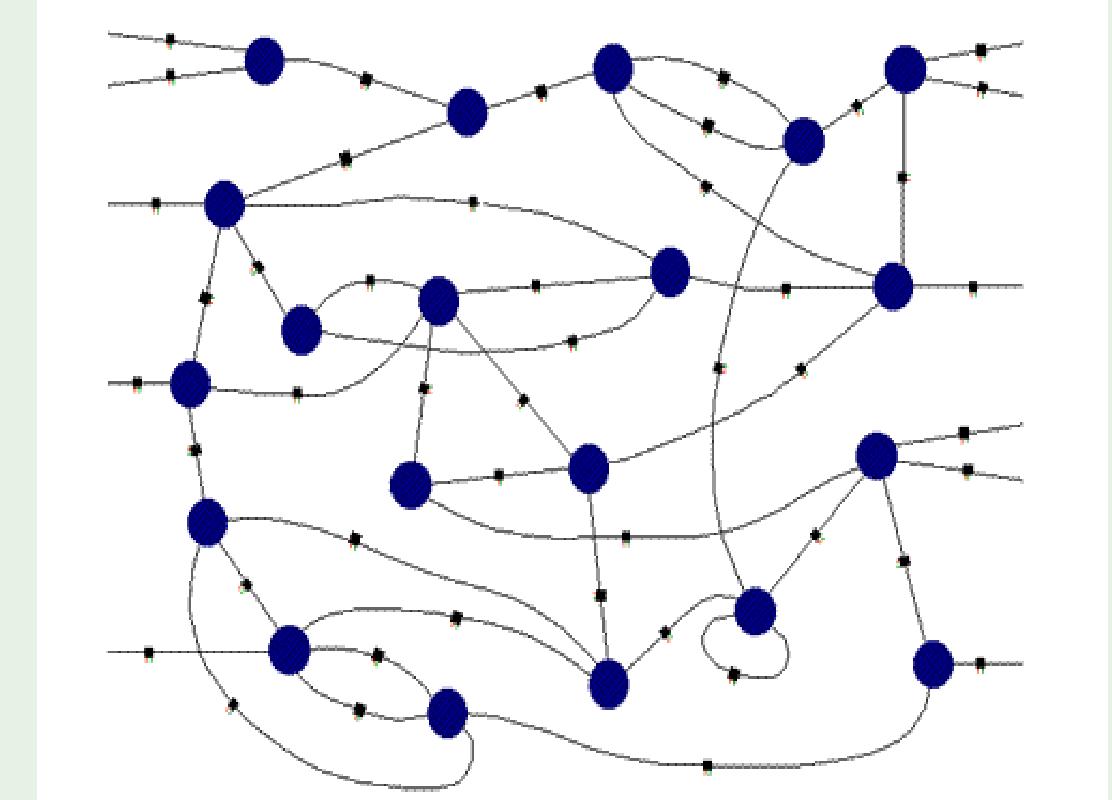
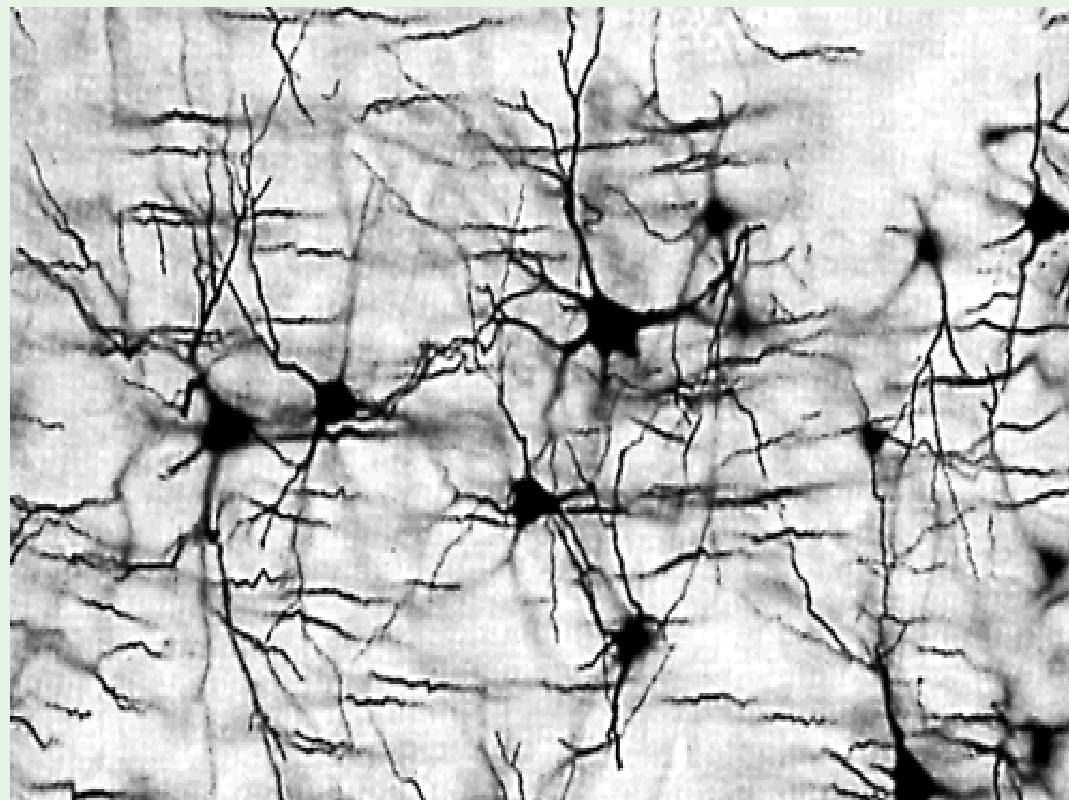
- Stochastic gradient descent
- Neural network model
- Backpropagation algorithm

Biological inspiration

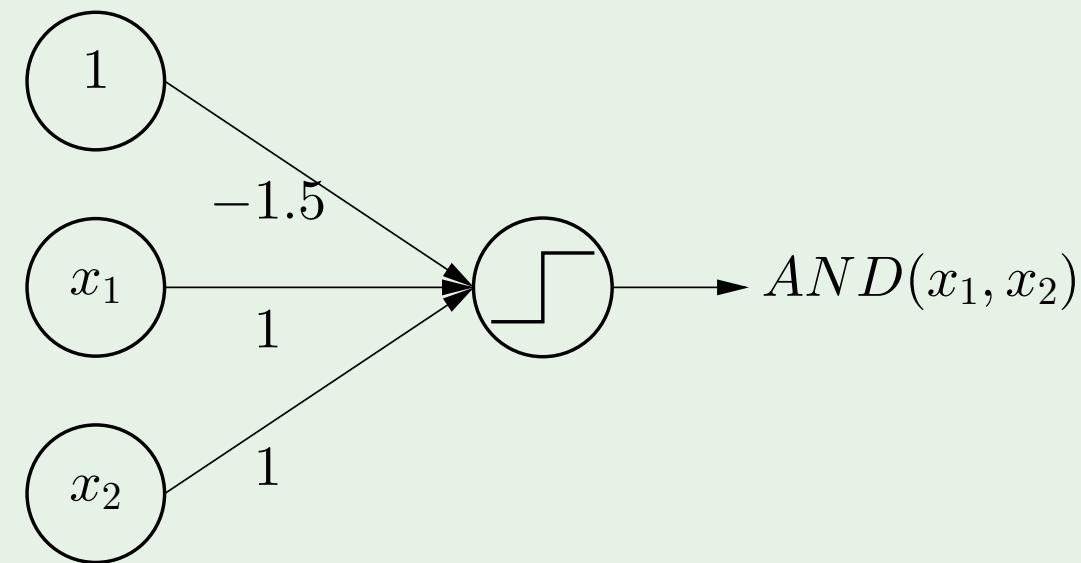
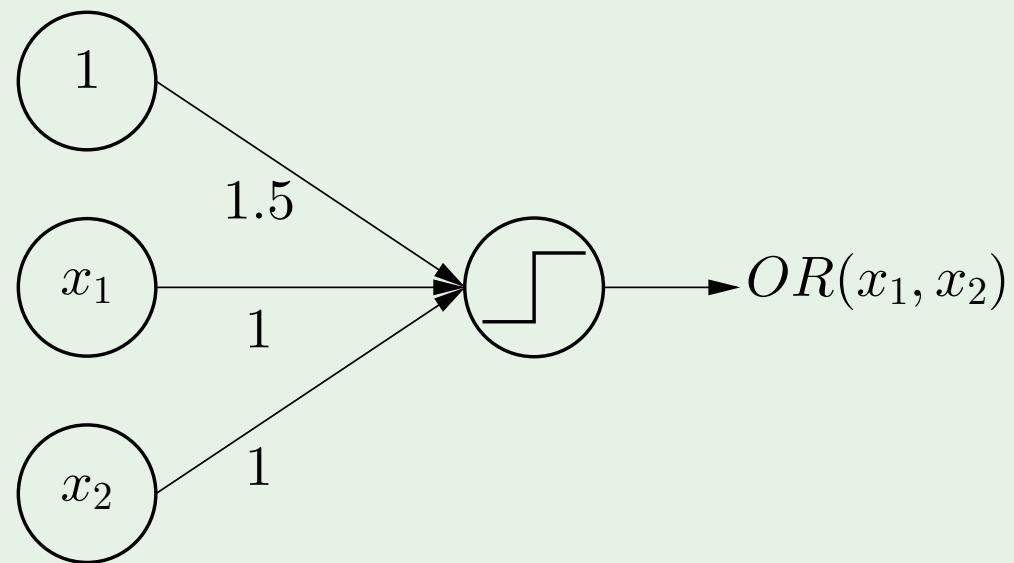
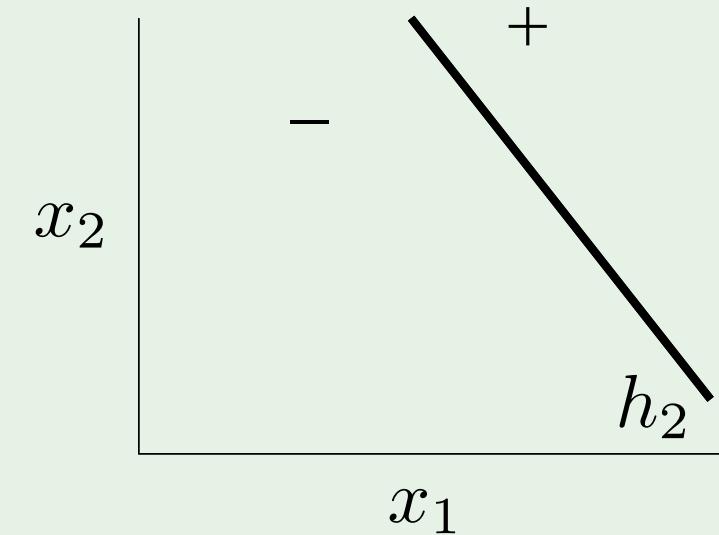
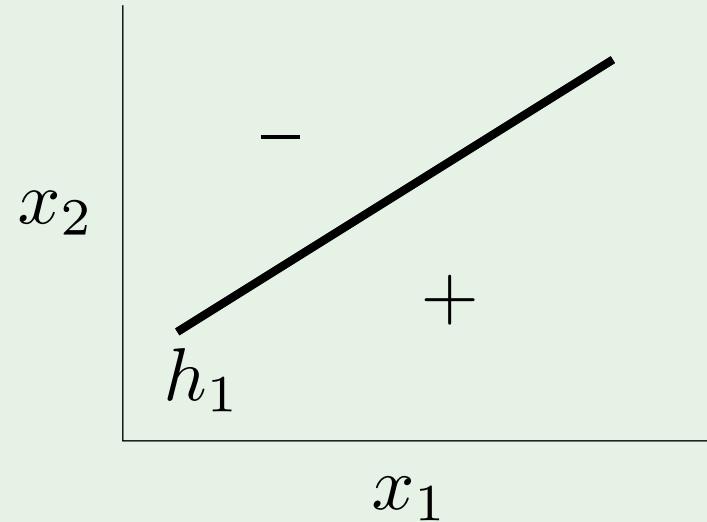
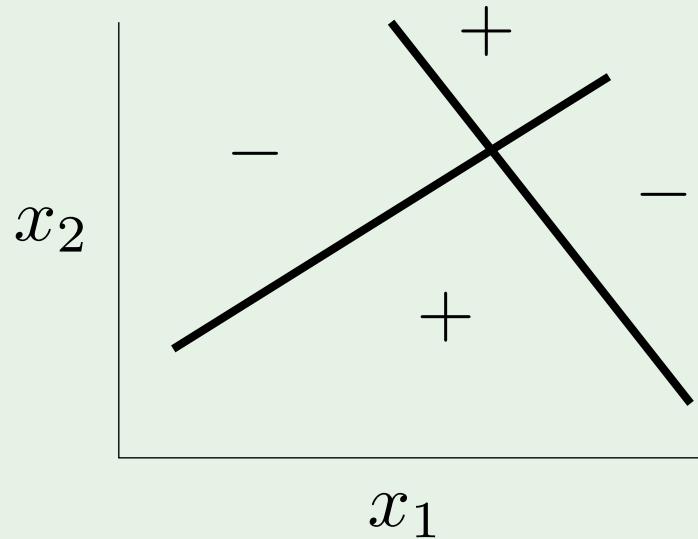
biological function



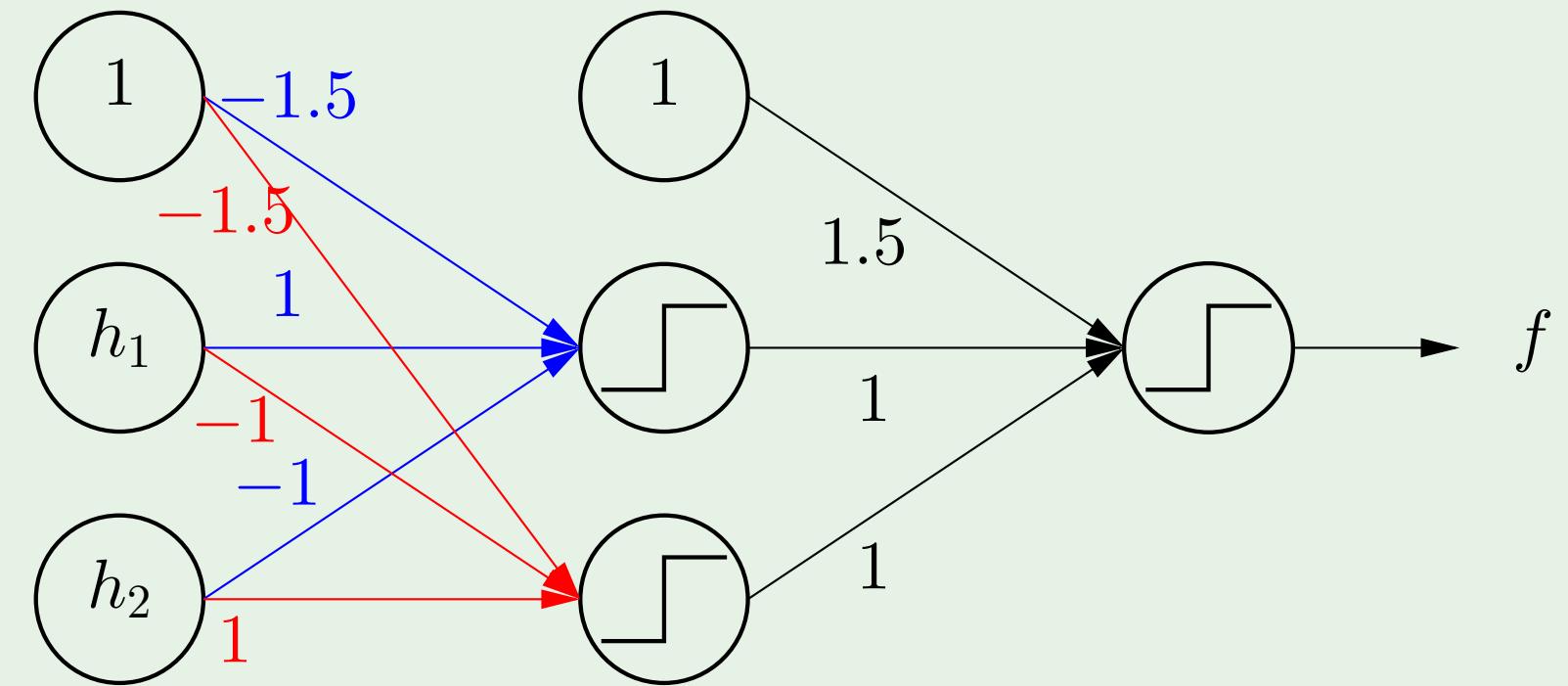
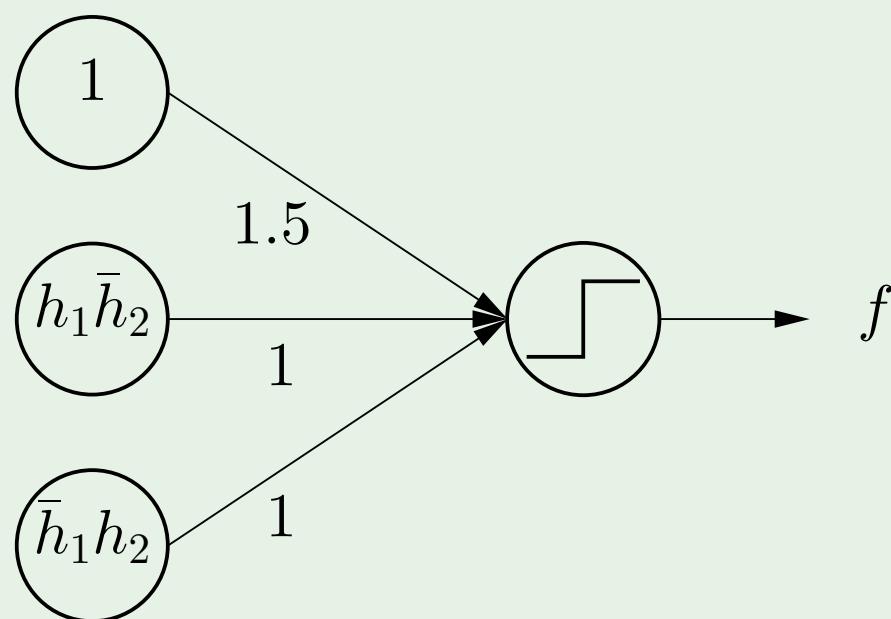
biological structure



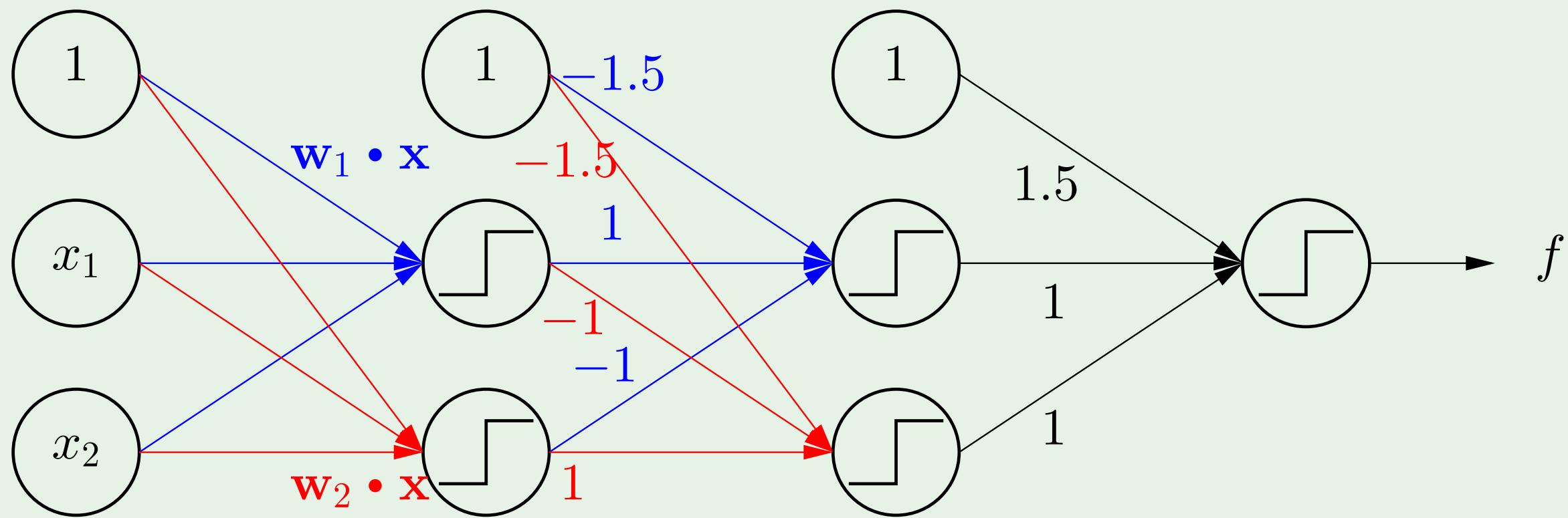
Combining perceptrons



Creating layers

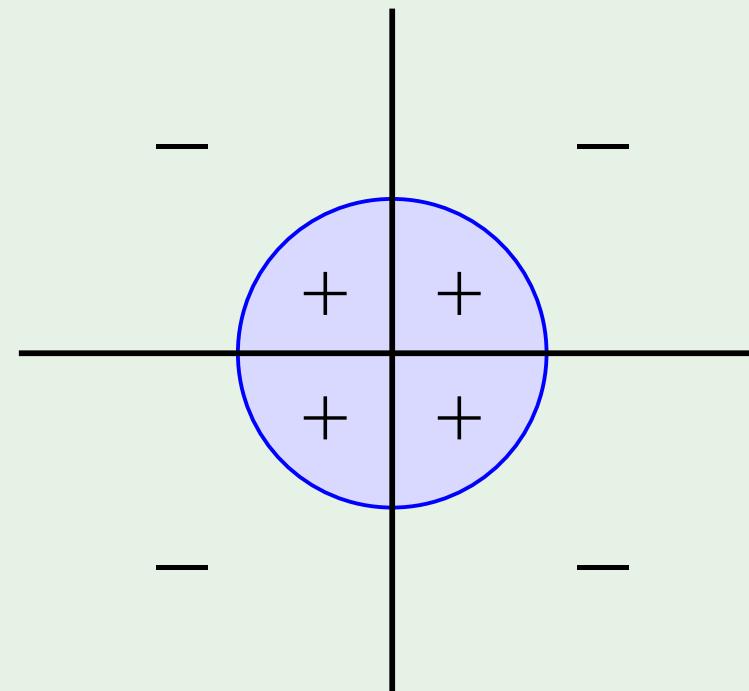


The multilayer perceptron

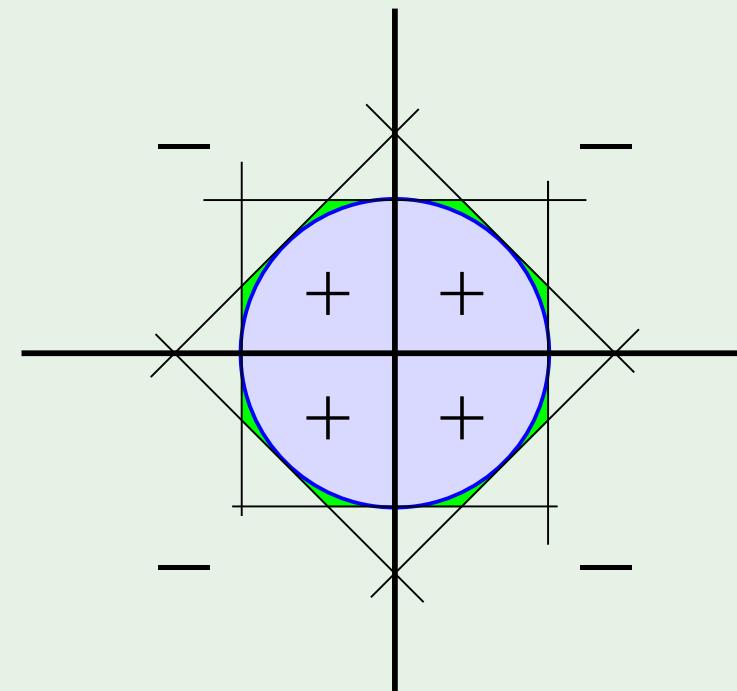


3 layers “feedforward”

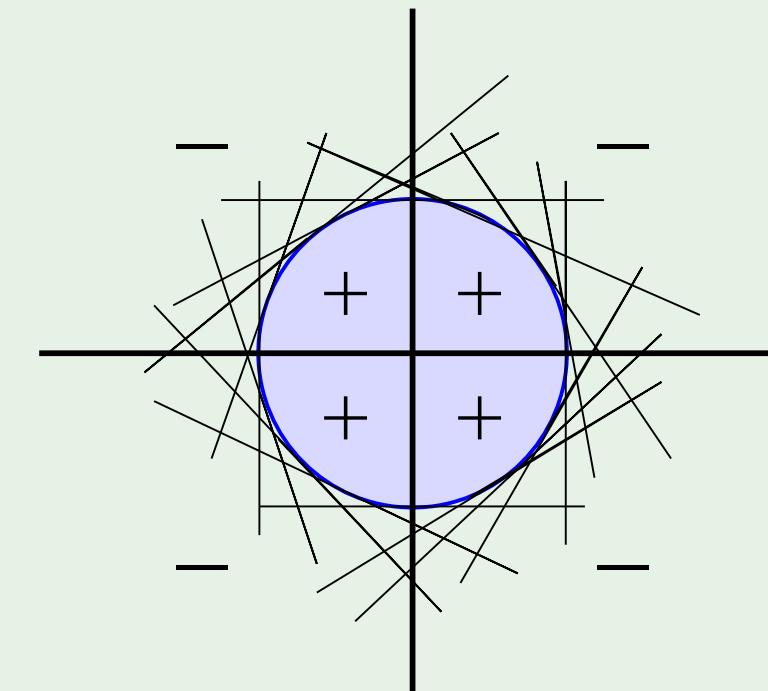
A powerful model



Target



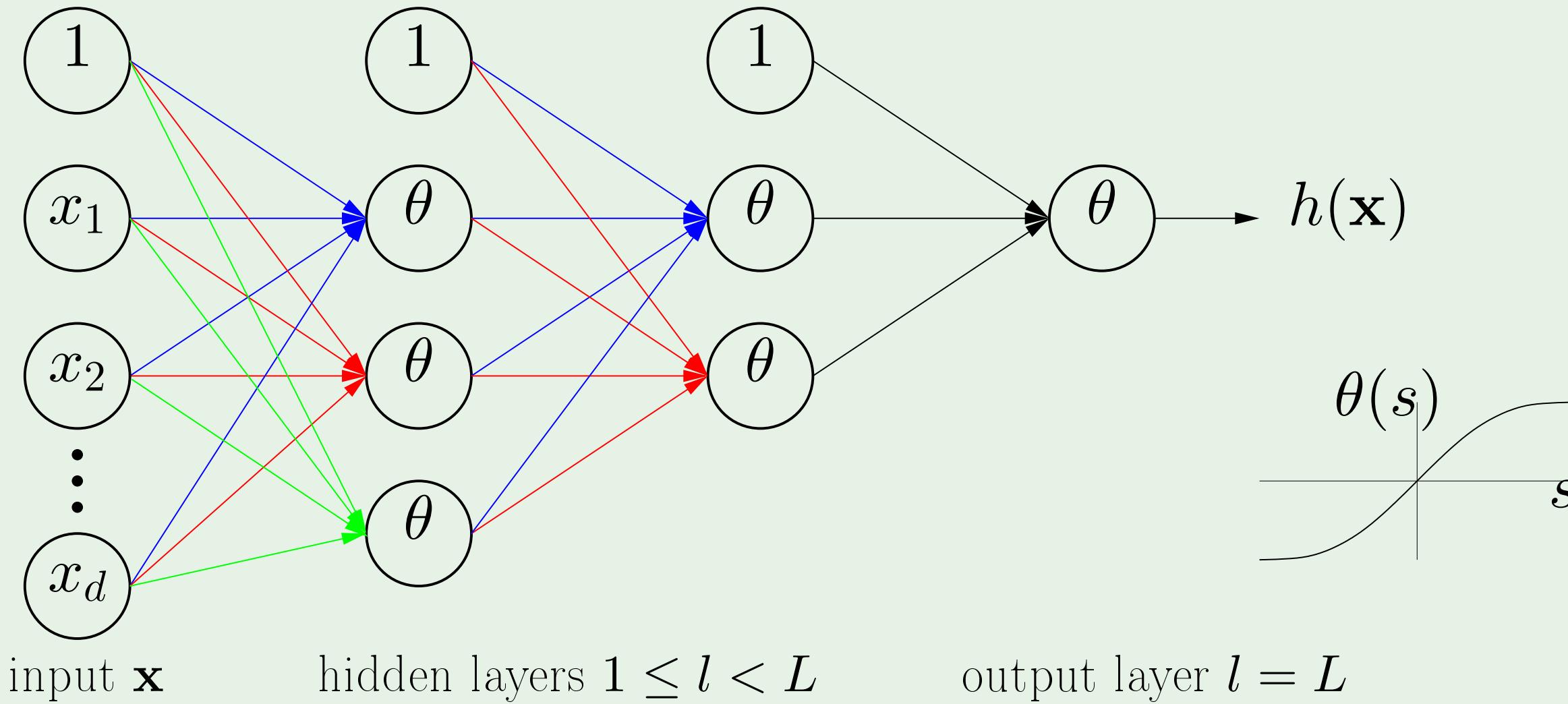
8 perceptrons



16 perceptrons

2 red flags for generalization and optimization

The neural network

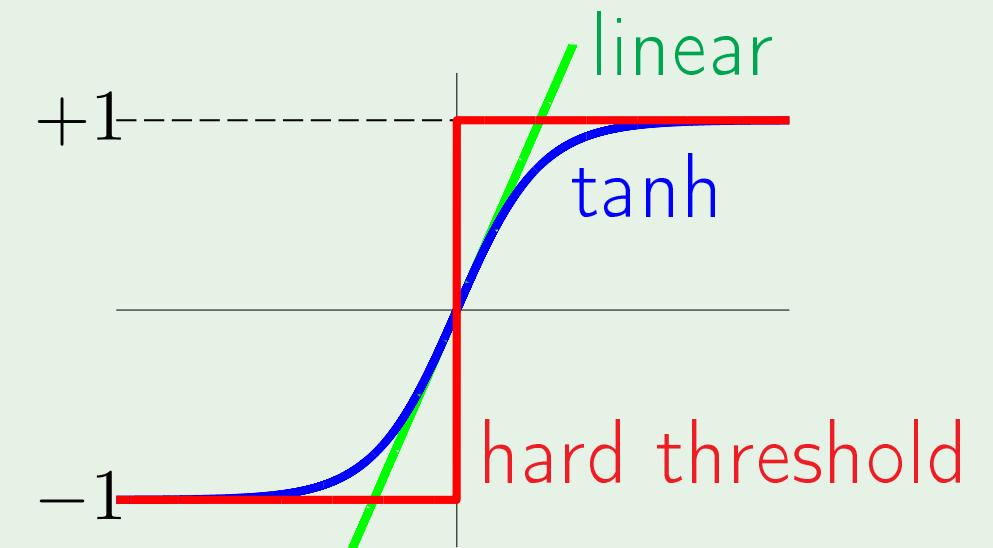


How the network operates

$$w_{ij}^{(l)} \quad \begin{cases} 1 \leq l \leq L & \text{layers} \\ 0 \leq i \leq d^{(l-1)} & \text{inputs} \\ 1 \leq j \leq d^{(l)} & \text{outputs} \end{cases}$$

$$x_j^{(l)} = \theta(s_j^{(l)}) = \theta \left(\sum_{i=0}^{d^{(l-1)}} w_{ij}^{(l)} x_i^{(l-1)} \right)$$

Apply \mathbf{x} to $x_1^{(0)} \dots x_{d^{(0)}}^{(0)} \rightarrow \rightarrow x_1^{(L)} = h(\mathbf{x})$



$$\theta(s) = \tanh(s) = \frac{e^s - e^{-s}}{e^s + e^{-s}}$$

Outline

- Stochastic gradient descent
- Neural network model
- Backpropagation algorithm

Applying SGD

All the weights $\mathbf{w} = \{w_{ij}^{(l)}\}$ determine $h(\mathbf{x})$

Error on example (\mathbf{x}_n, y_n) is

$$\mathbf{e}(h(\mathbf{x}_n), y_n) = \mathbf{e}(\mathbf{w})$$

To implement SGD, we need the gradient

$$\nabla \mathbf{e}(\mathbf{w}): \frac{\partial \mathbf{e}(\mathbf{w})}{\partial w_{ij}^{(l)}} \quad \text{for all } i, j, l$$

Computing $\frac{\partial \text{e}(\mathbf{w})}{\partial w_{ij}^{(l)}}$

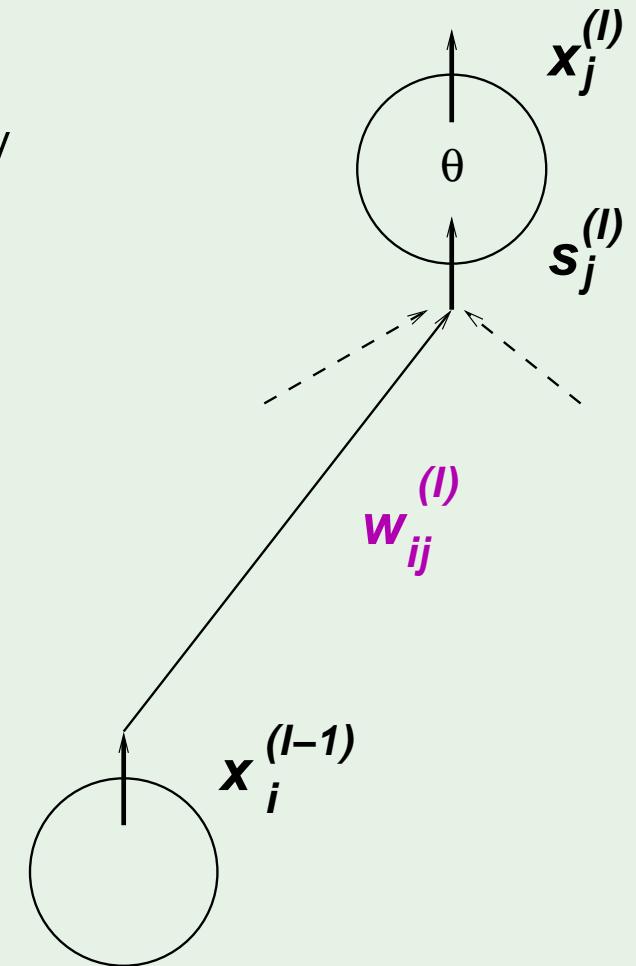
We can evaluate $\frac{\partial \text{e}(\mathbf{w})}{\partial w_{ij}^{(l)}}$ one by one: analytically or numerically

A trick for efficient computation:

$$\frac{\partial \text{e}(\mathbf{w})}{\partial w_{ij}^{(l)}} = \frac{\partial \text{e}(\mathbf{w})}{\partial s_j^{(l)}} \times \frac{\partial s_j^{(l)}}{\partial w_{ij}^{(l)}}$$

We have $\frac{\partial s_j^{(l)}}{\partial w_{ij}^{(l)}} = x_i^{(l-1)}$

We only need: $\frac{\partial \text{e}(\mathbf{w})}{\partial s_j^{(l)}} = \delta_j^{(l)}$



δ for the final layer

$$\delta_j^{(l)} = \frac{\partial \mathbf{e}(\mathbf{w})}{\partial s_j^{(l)}}$$

For the final layer $l = L$ and $j = 1$:

$$\delta_1^{(L)} = \frac{\partial \mathbf{e}(\mathbf{w})}{\partial s_1^{(L)}}$$

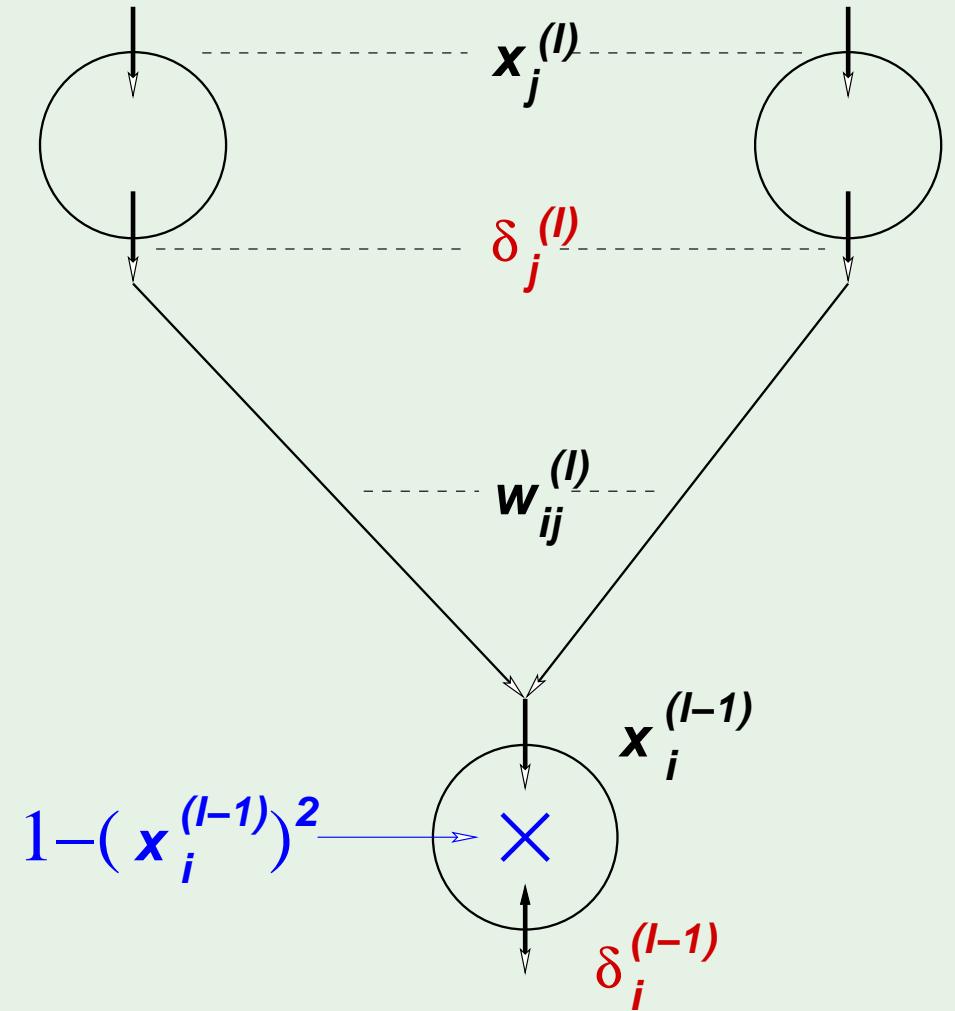
$$\mathbf{e}(\mathbf{w}) = (x_1^{(L)} - y_n)^2$$

$$x_1^{(L)} = \theta(s_1^{(L)})$$

$$\theta'(s) = 1 - \theta^2(s) \quad \text{for the tanh}$$

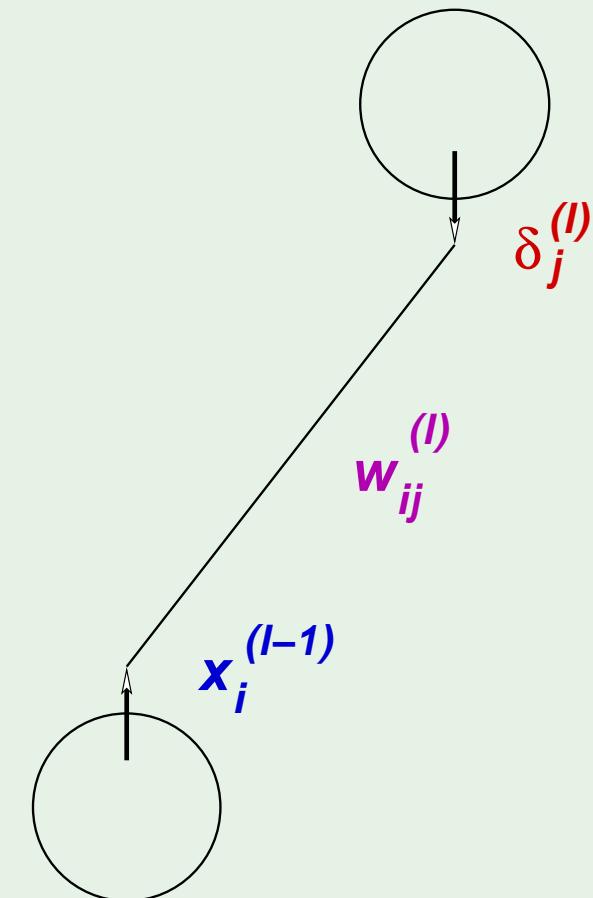
Back propagation of δ

$$\begin{aligned}
 \delta_i^{(l-1)} &= \frac{\partial \mathbf{e}(\mathbf{w})}{\partial s_i^{(l-1)}} \\
 &= \sum_{j=1}^{d^{(l)}} \frac{\partial \mathbf{e}(\mathbf{w})}{\partial s_j^{(l)}} \times \frac{\partial s_j^{(l)}}{\partial x_i^{(l-1)}} \times \frac{\partial x_i^{(l-1)}}{\partial s_i^{(l-1)}} \\
 &= \sum_{j=1}^{d^{(l)}} \delta_j^{(l)} \times w_{ij}^{(l)} \times \theta'(s_i^{(l-1)}) \\
 \delta_i^{(l-1)} &= (1 - (x_i^{(l-1)})^2) \sum_{j=1}^{d^{(l)}} w_{ij}^{(l)} \delta_j^{(l)}
 \end{aligned}$$



Backpropagation algorithm

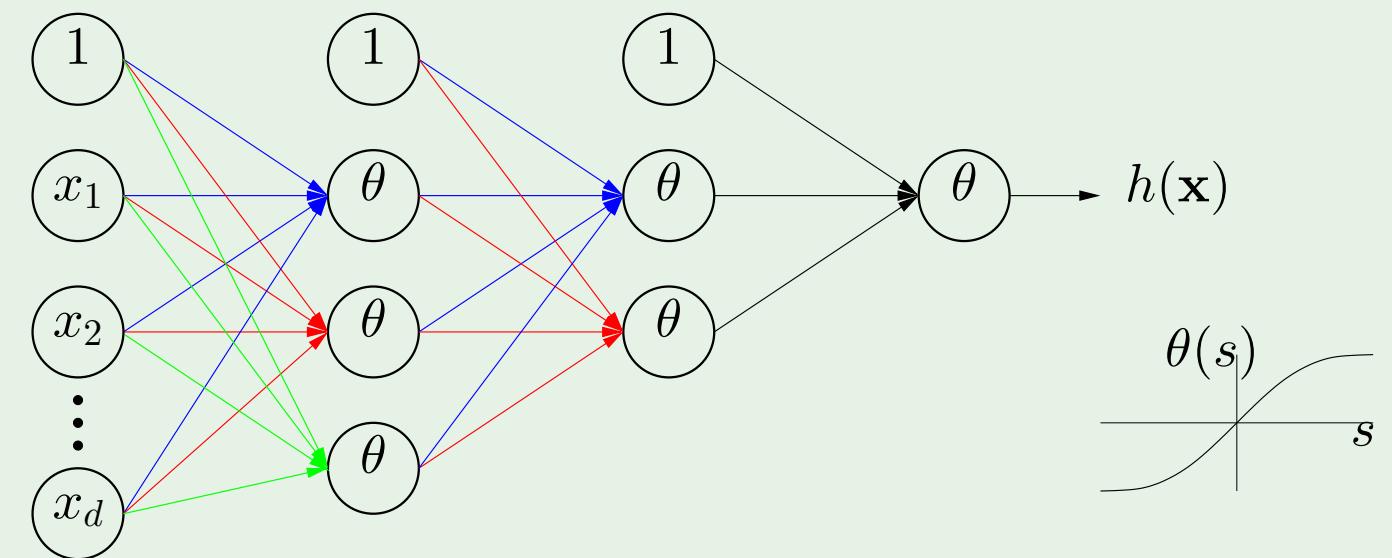
- 1: Initialize all weights $w_{ij}^{(l)}$ **at random**
- 2: **for** $t = 0, 1, 2, \dots$ **do**
- 3: Pick $n \in \{1, 2, \dots, N\}$
- 4: *Forward:* Compute all $x_j^{(l)}$
- 5: *Backward:* Compute all $\delta_j^{(l)}$
- 6: Update the weights: $w_{ij}^{(l)} \leftarrow w_{ij}^{(l)} - \eta x_i^{(l-1)} \delta_j^{(l)}$
- 7: Iterate to the next step until it is time to stop
- 8: Return the final weights $w_{ij}^{(l)}$



Final remark: hidden layers

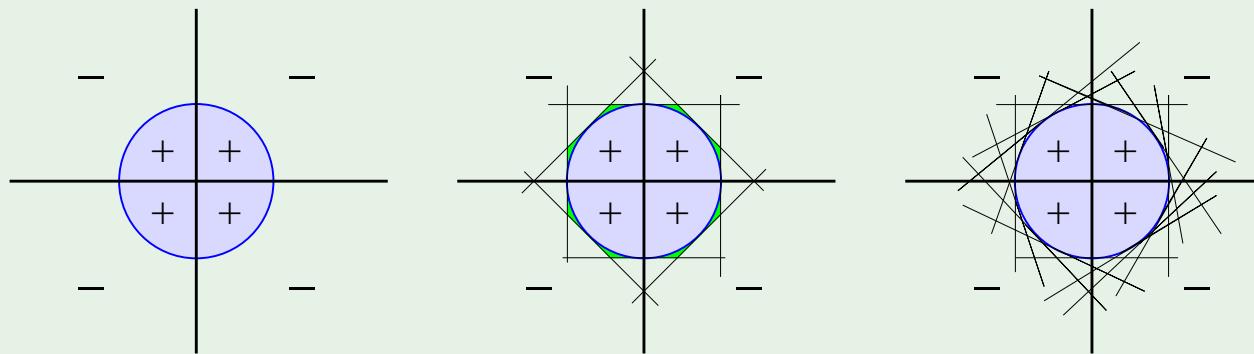
learned nonlinear transform

interpretation?



Review of Lecture 10

- Multilayer perceptrons

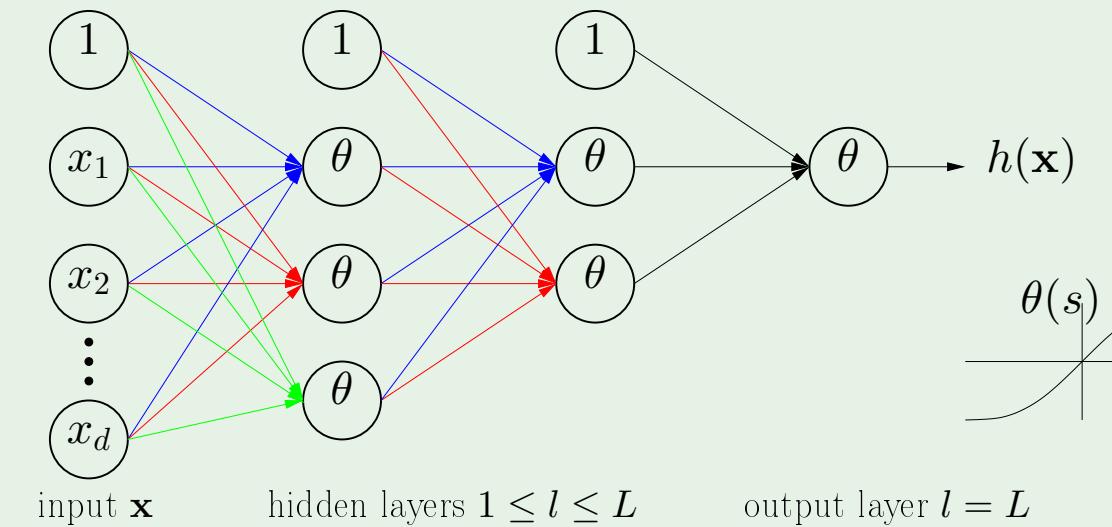


Logical combinations of perceptrons

- Neural networks

$$x_j^{(l)} = \theta \left(\sum_{i=0}^{d^{(l-1)}} w_{ij}^{(l)} x_i^{(l-1)} \right)$$

where $\theta(s) = \tanh(s)$



- Backpropagation

$$\Delta w_{ij}^{(l)} = -\eta x_i^{(l-1)} \delta_j^{(l)}$$

where

$$\delta_i^{(l-1)} = (1 - (x_i^{(l-1)})^2) \sum_{j=1}^{d^{(l)}} w_{ij}^{(l)} \delta_j^{(l)}$$

Learning From Data

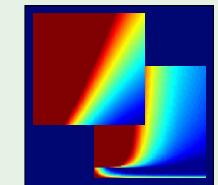
Yaser S. Abu-Mostafa
California Institute of Technology

Lecture 11: Overfitting



Sponsored by Caltech's Provost Office, E&AS Division, and IST

• Tuesday, May 8, 2012



Outline

- What is overfitting?
- The role of noise
- Deterministic noise
- Dealing with overfitting

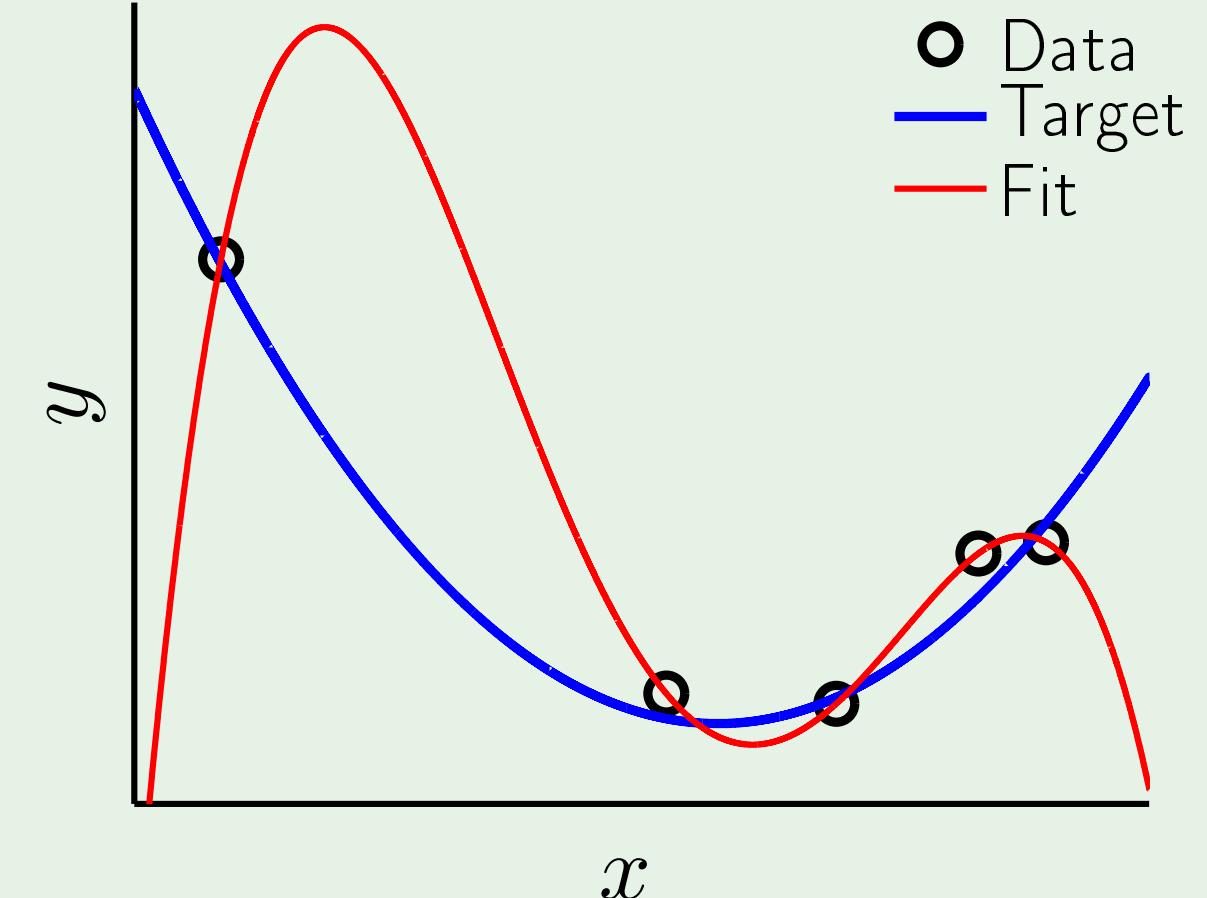
Illustration of overfitting

Simple target function

5 data points- **noisy**

4th-order polynomial fit

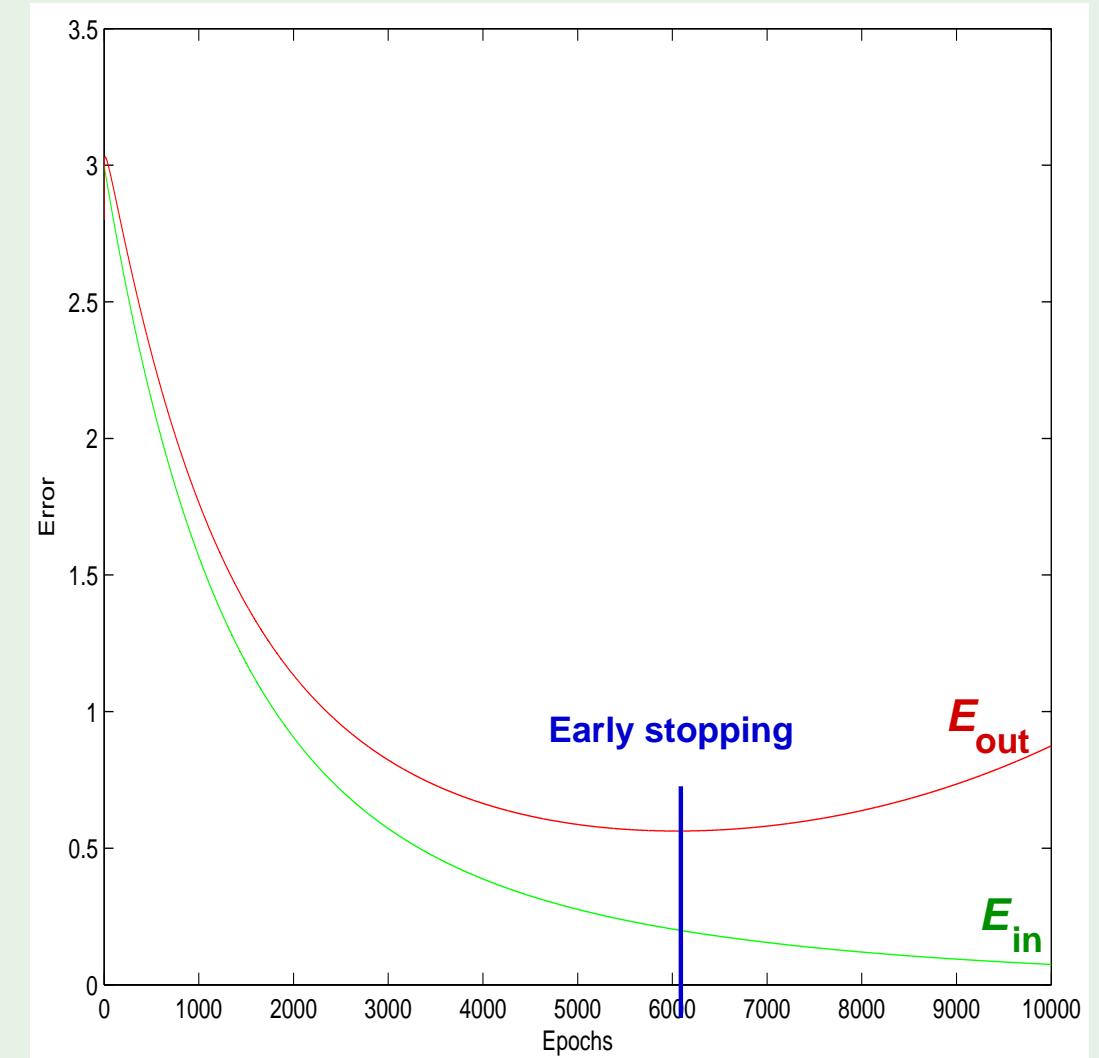
$E_{\text{in}} = 0, E_{\text{out}}$ is huge



Overfitting versus bad generalization

Neural network fitting noisy data

Overfitting: $E_{\text{in}} \downarrow$ $E_{\text{out}} \uparrow$



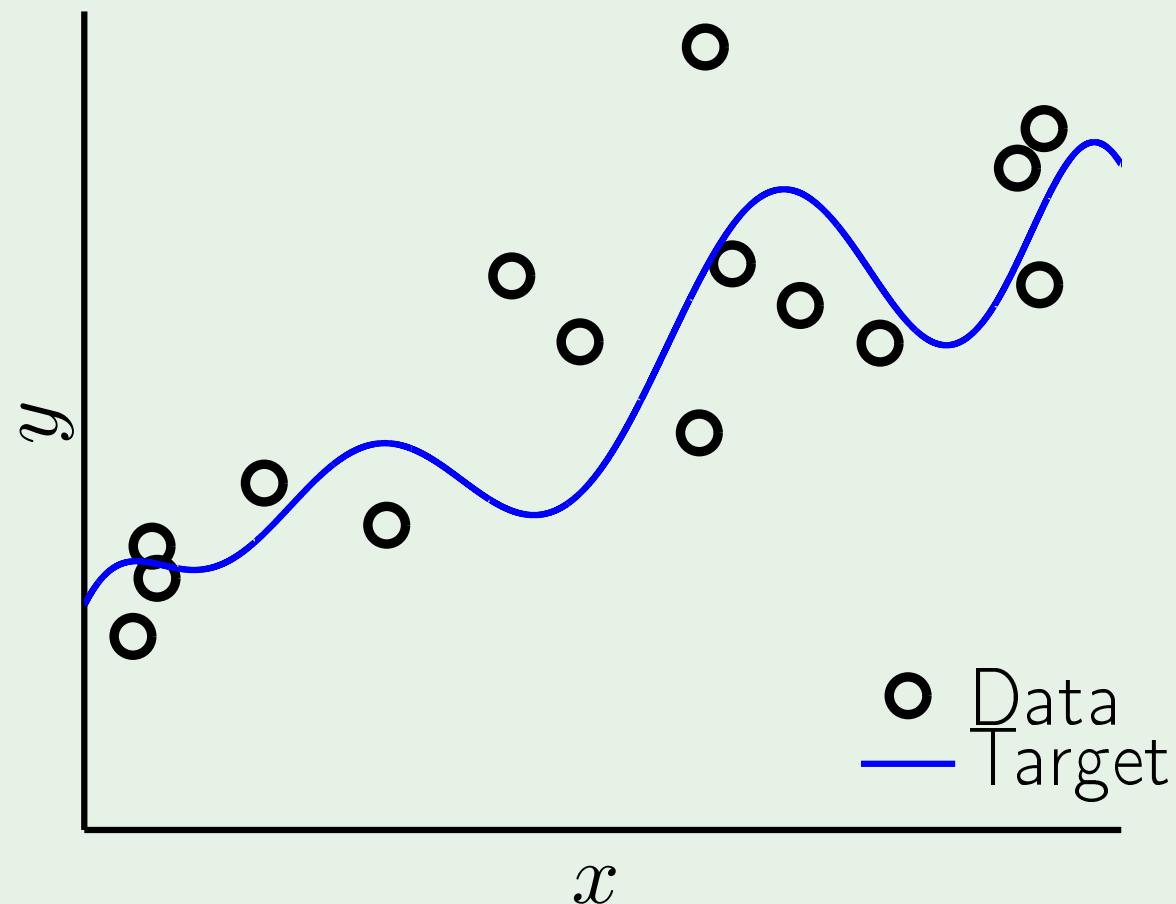
The culprit

Overfitting: “fitting the data more than is warranted”

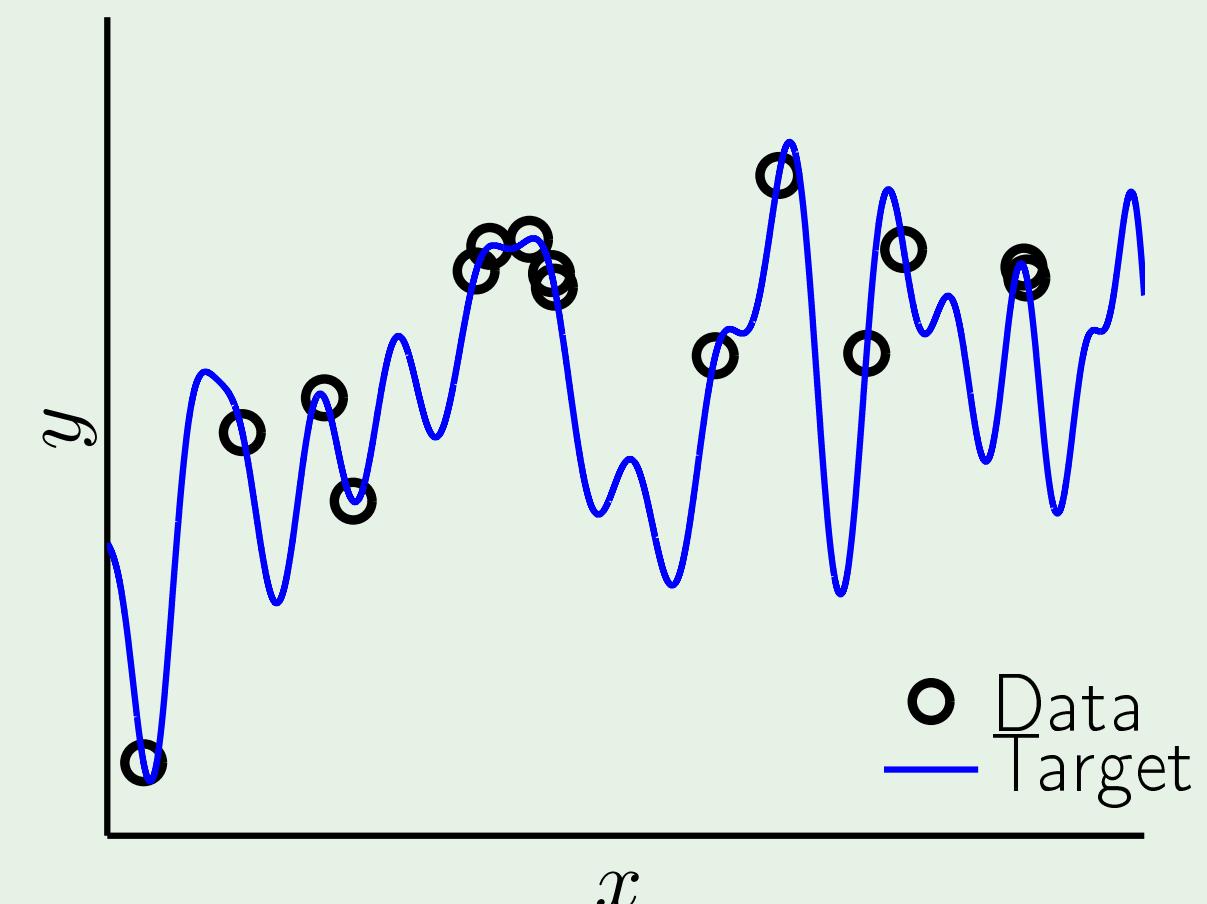
Culprit: fitting the noise - **harmful**

Case study

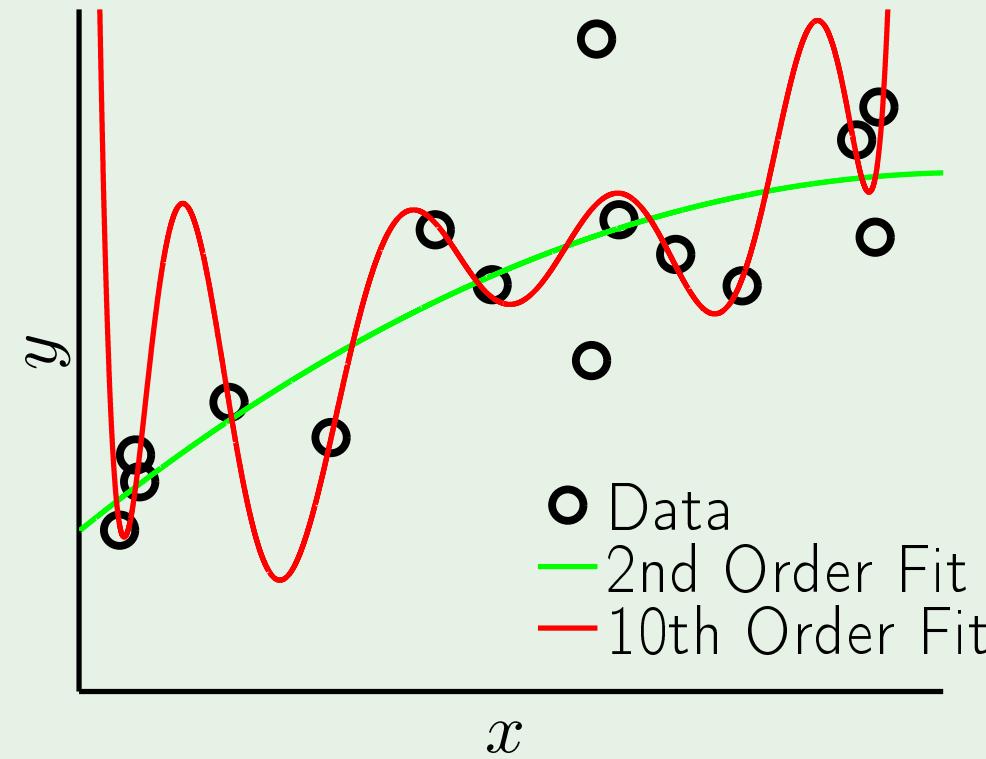
10th-order target + noise



50th-order target

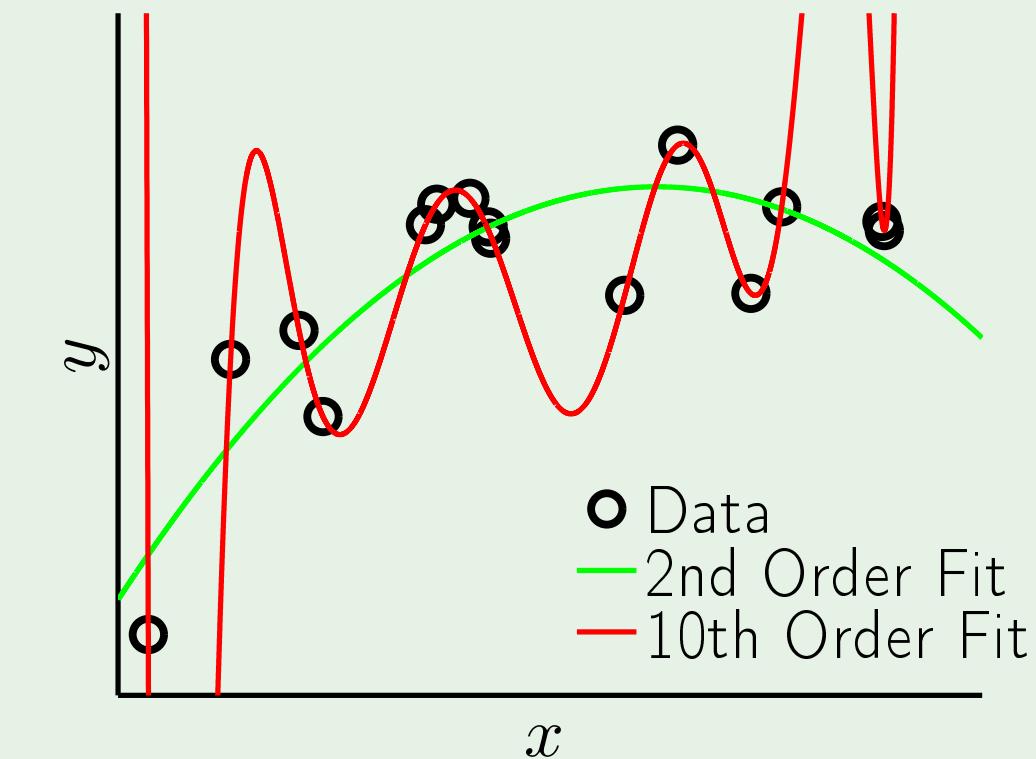


Two fits for each target



Noisy low-order target

	2nd Order	10th Order
E_{in}	0.050	0.034
E_{out}	0.127	9.00



Noiseless high-order target

	2nd Order	10th Order
E_{in}	0.029	10^{-5}
E_{out}	0.120	7680

An irony of two learners

Two learners O and R

They know the target is 10th order

O chooses \mathcal{H}_{10}

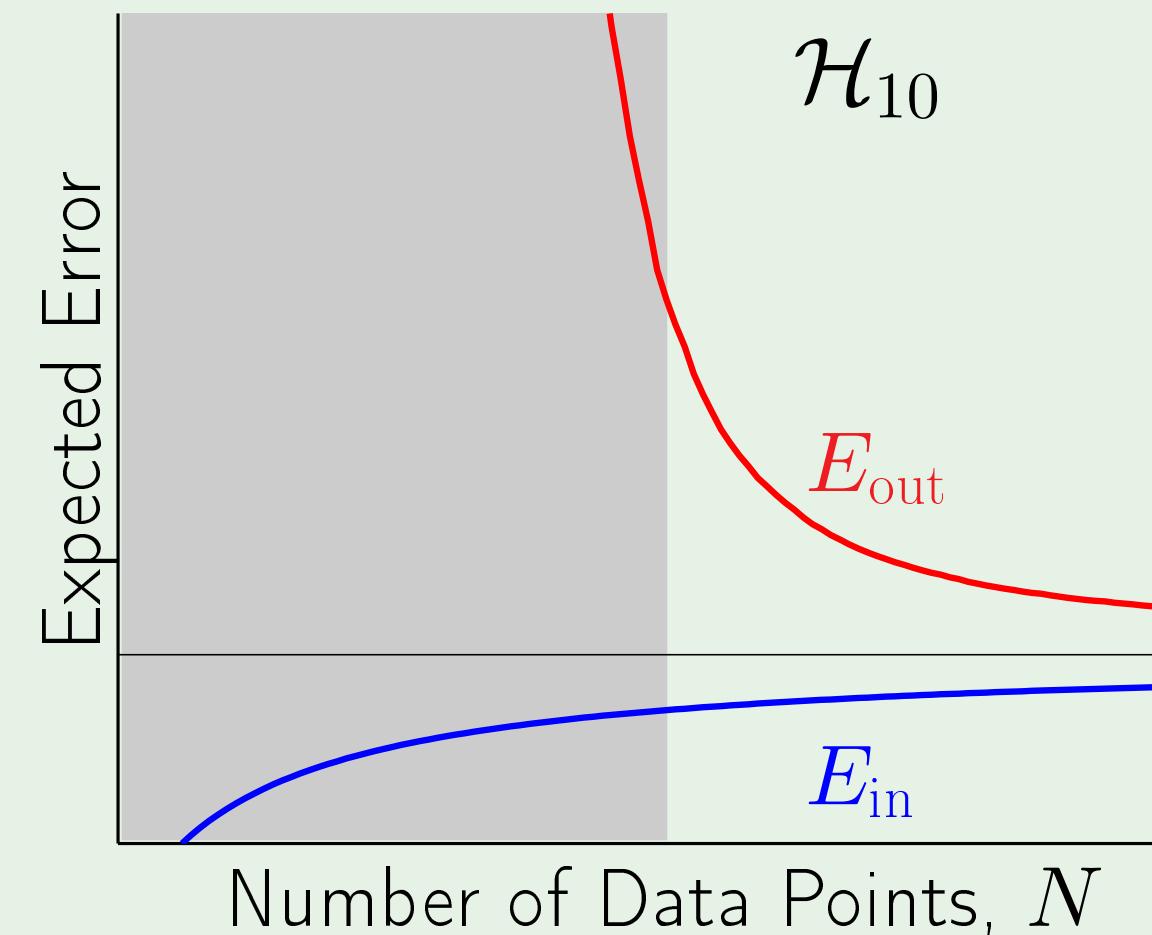
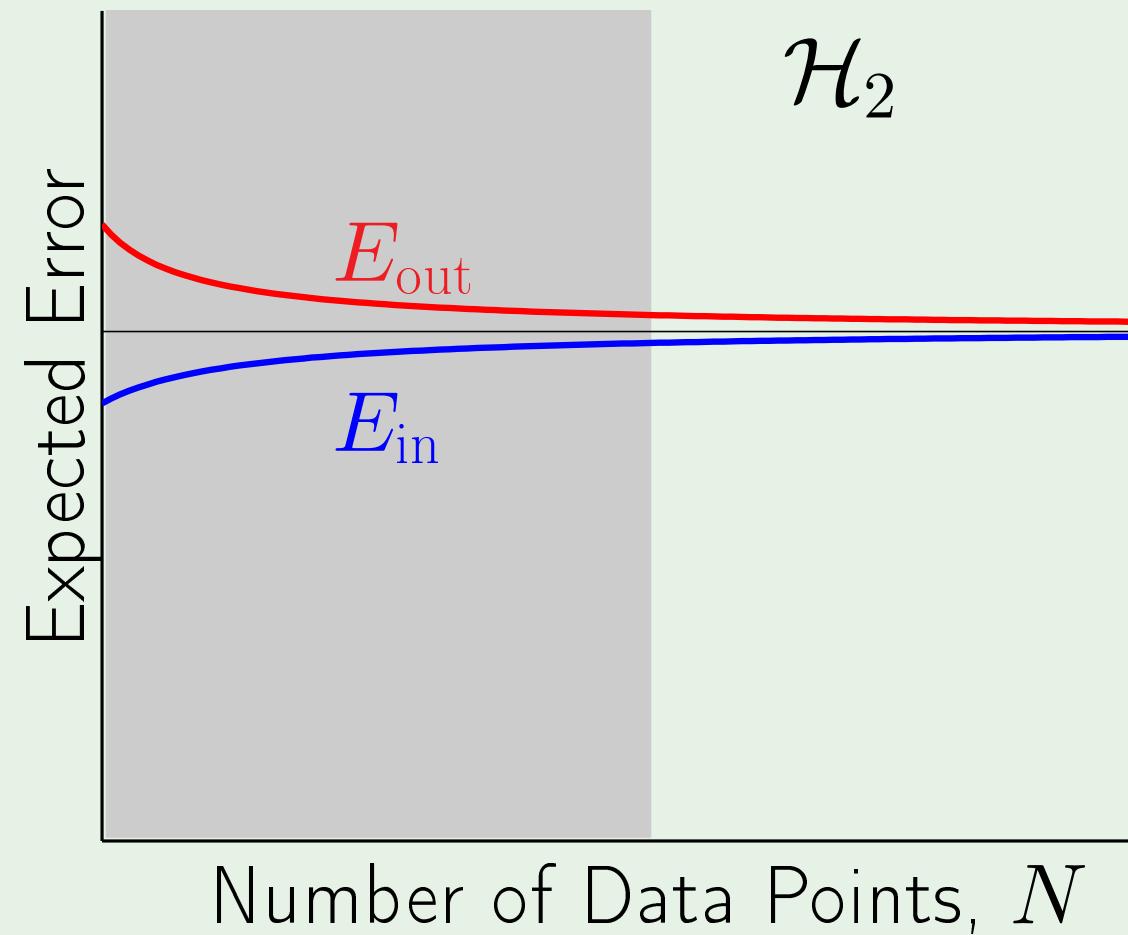
R chooses \mathcal{H}_2



Learning a 10th-order target

We have seen this case

Remember learning curves?

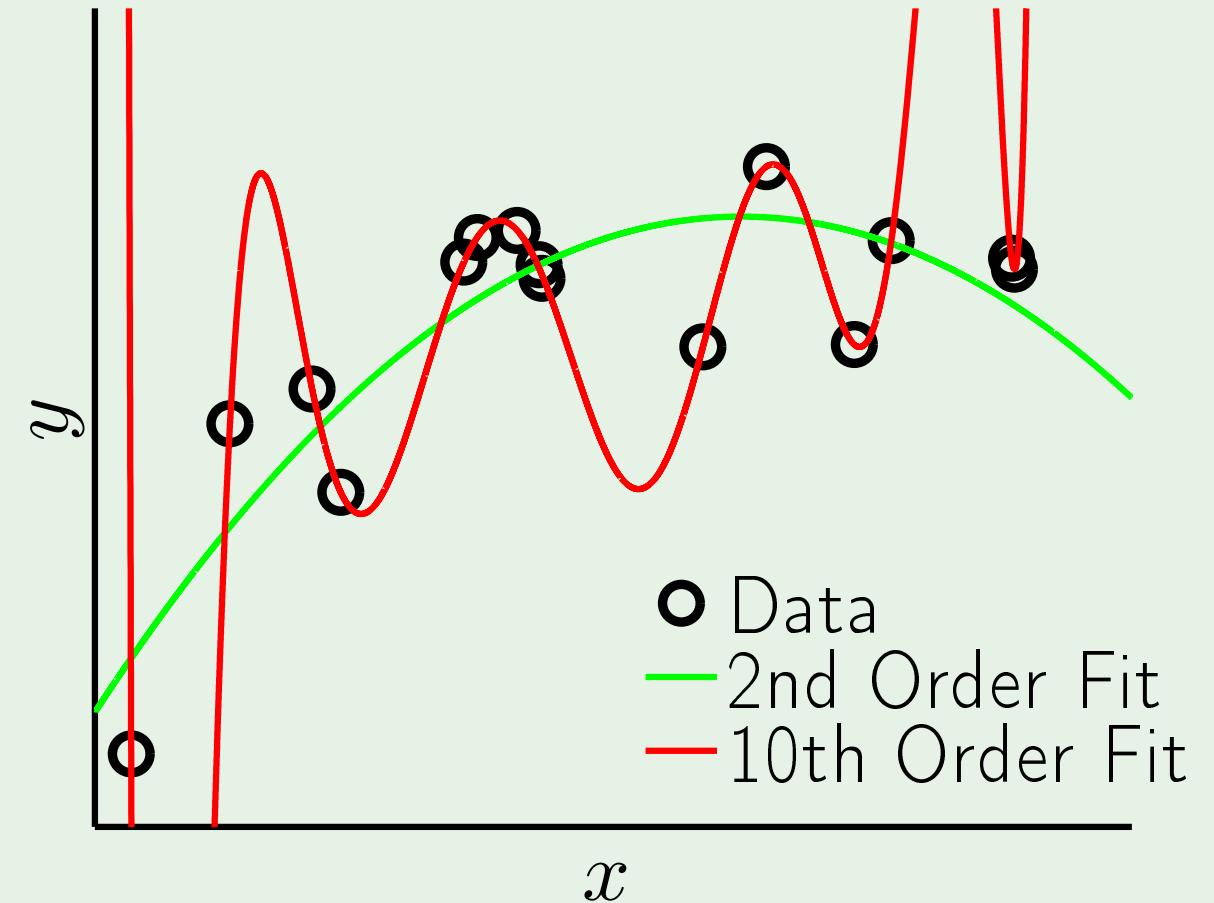


Even without noise

The two learners \mathcal{H}_{10} and \mathcal{H}_2

They know there is no noise

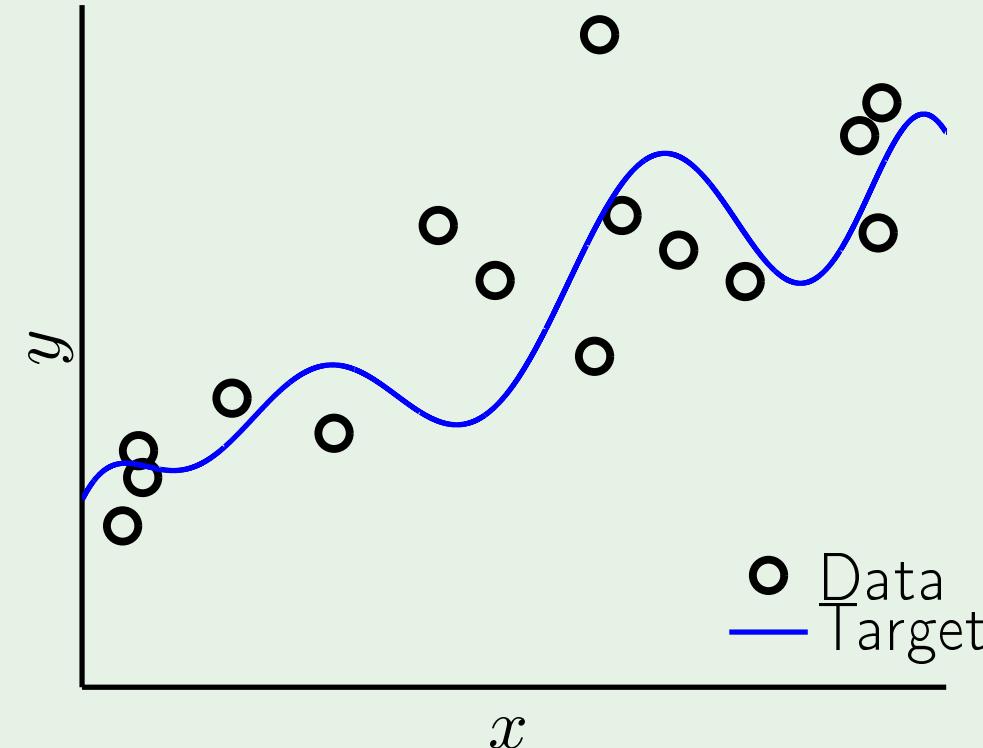
Is there really no noise?



Learning a 50th-order target

A detailed experiment

Impact of **noise level** and **target complexity**



$$y = f(x) + \underbrace{\epsilon(x)}_{\sigma^2} = \underbrace{\sum_{q=0}^{Q_f} \alpha_q x^q}_{\text{normalized}} + \epsilon(x)$$

noise level: σ^2

target complexity: Q_f

data set size: N

The overfit measure

We fit the data set $(x_1, y_1), \dots, (x_N, y_N)$ using our two models:

\mathcal{H}_2 : 2nd-order polynomials

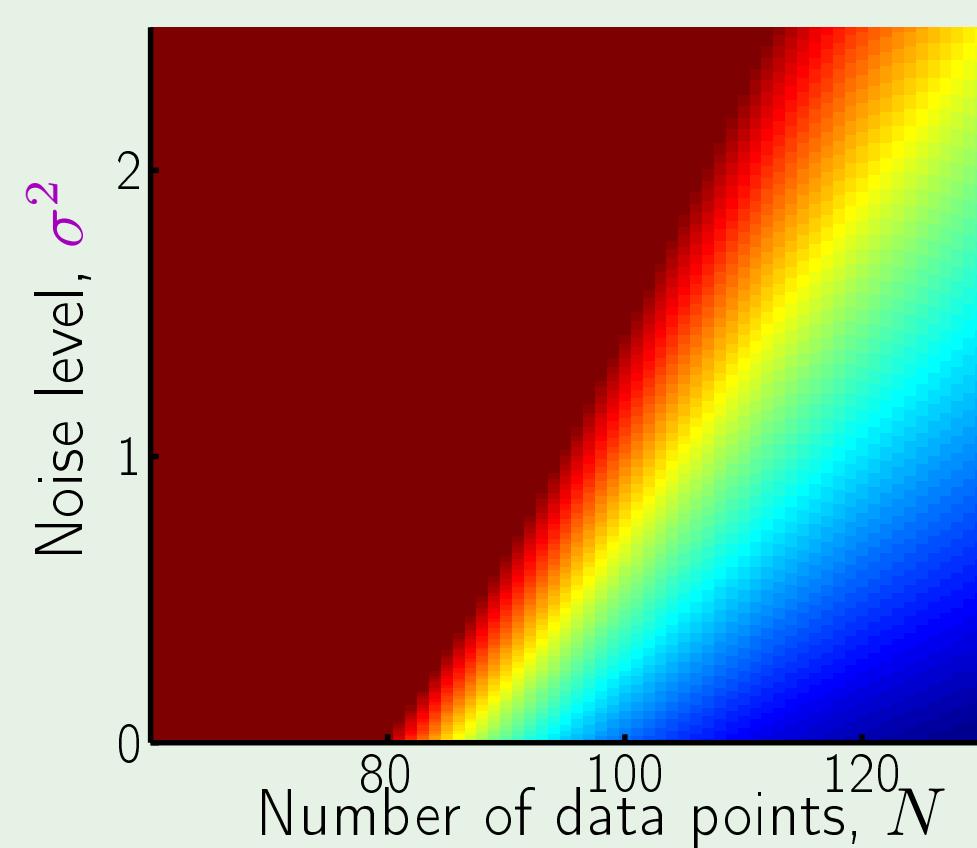
\mathcal{H}_{10} : 10th-order polynomials



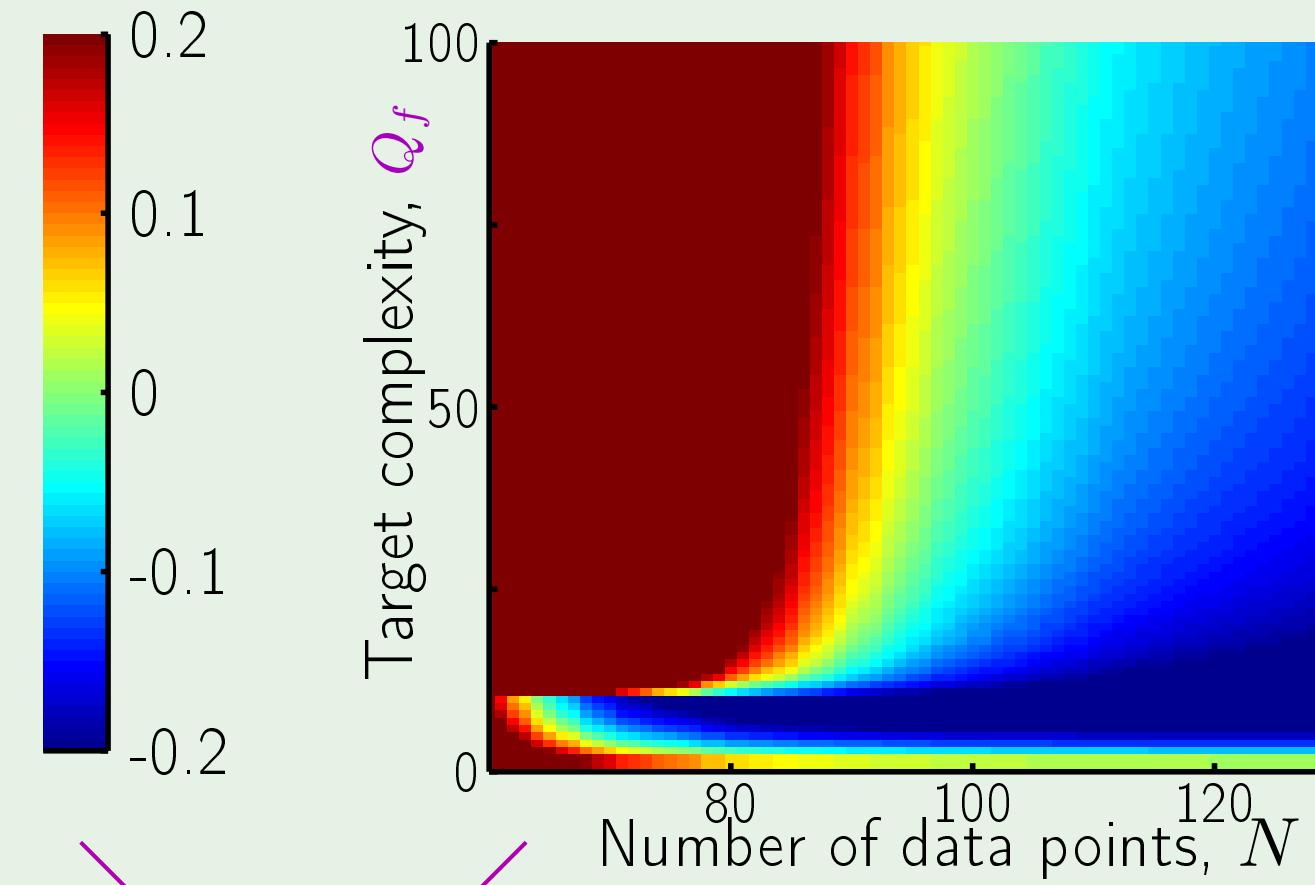
Compare out-of-sample errors of
 $g_2 \in \mathcal{H}_2$ and $g_{10} \in \mathcal{H}_{10}$

overfit measure: $E_{\text{out}}(g_{10}) - E_{\text{out}}(g_2)$

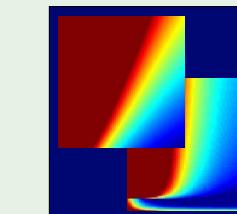
The results



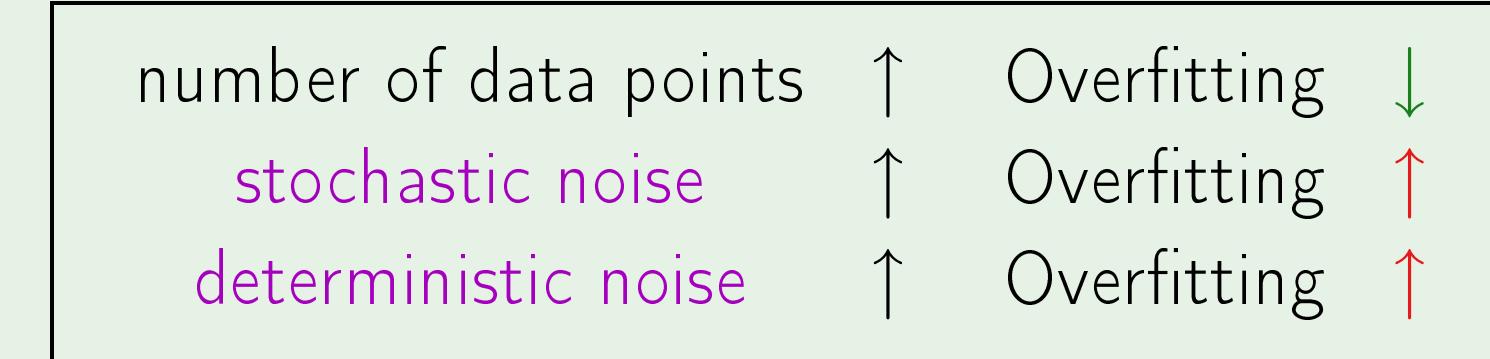
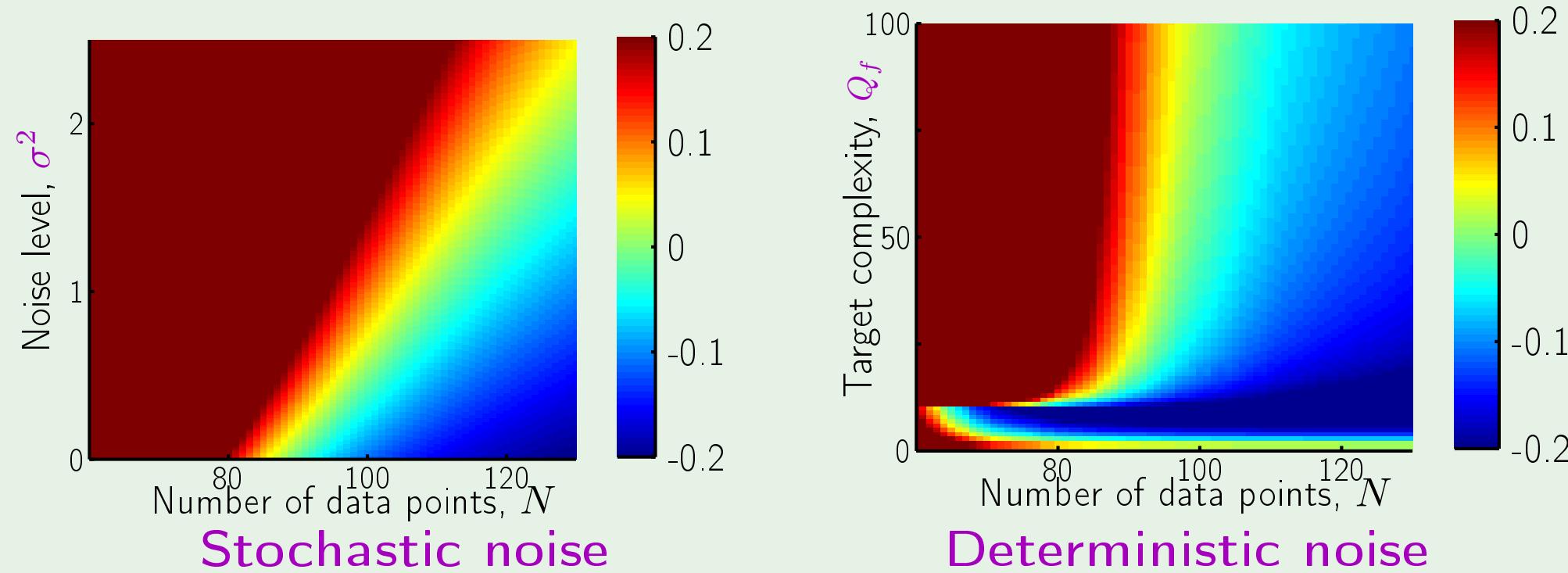
Impact of σ^2



Impact of Q_f



Impact of “noise”



Outline

- What is overfitting?
- The role of noise
- Deterministic noise
- Dealing with overfitting

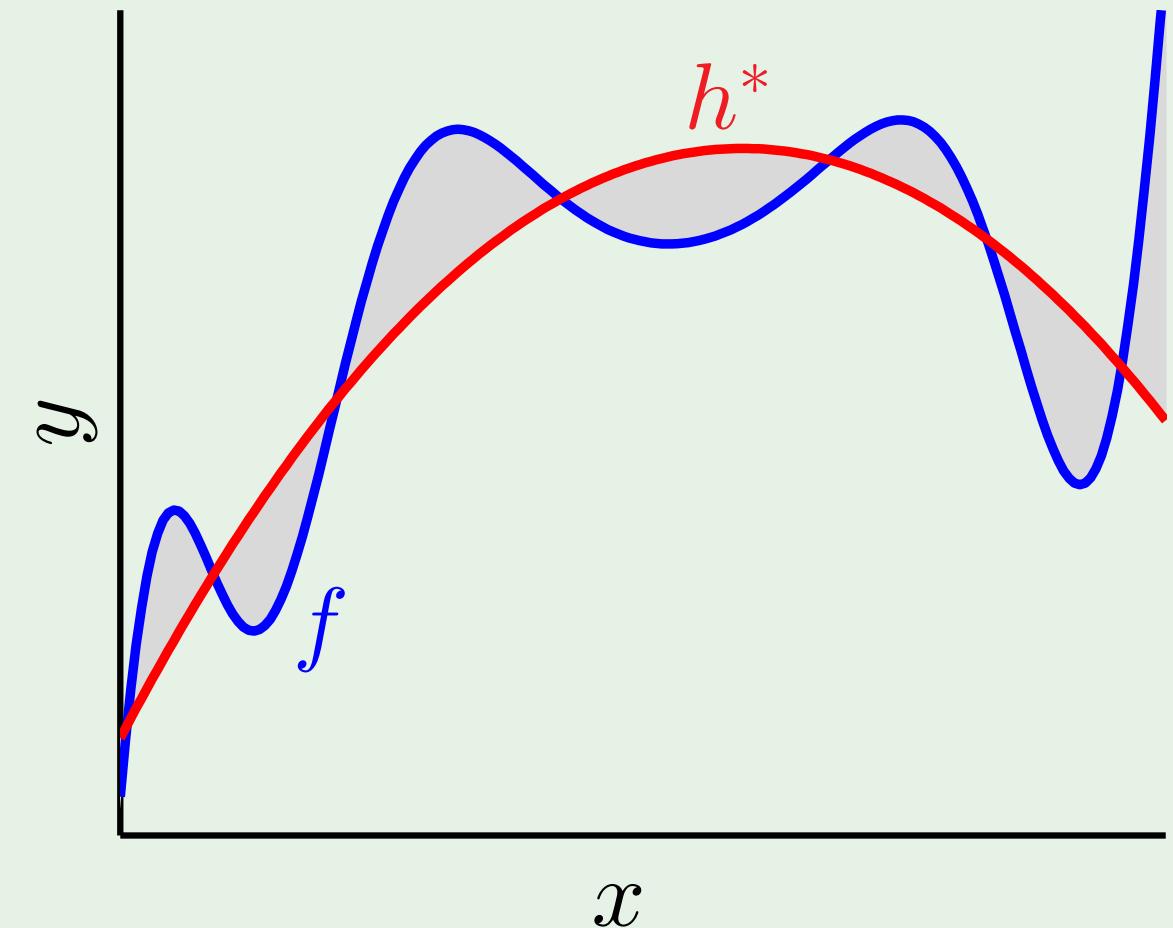
Definition of deterministic noise

The part of f that \mathcal{H} cannot capture: $f(\mathbf{x}) - h^*(\mathbf{x})$

Why “noise”?

Main differences with stochastic noise:

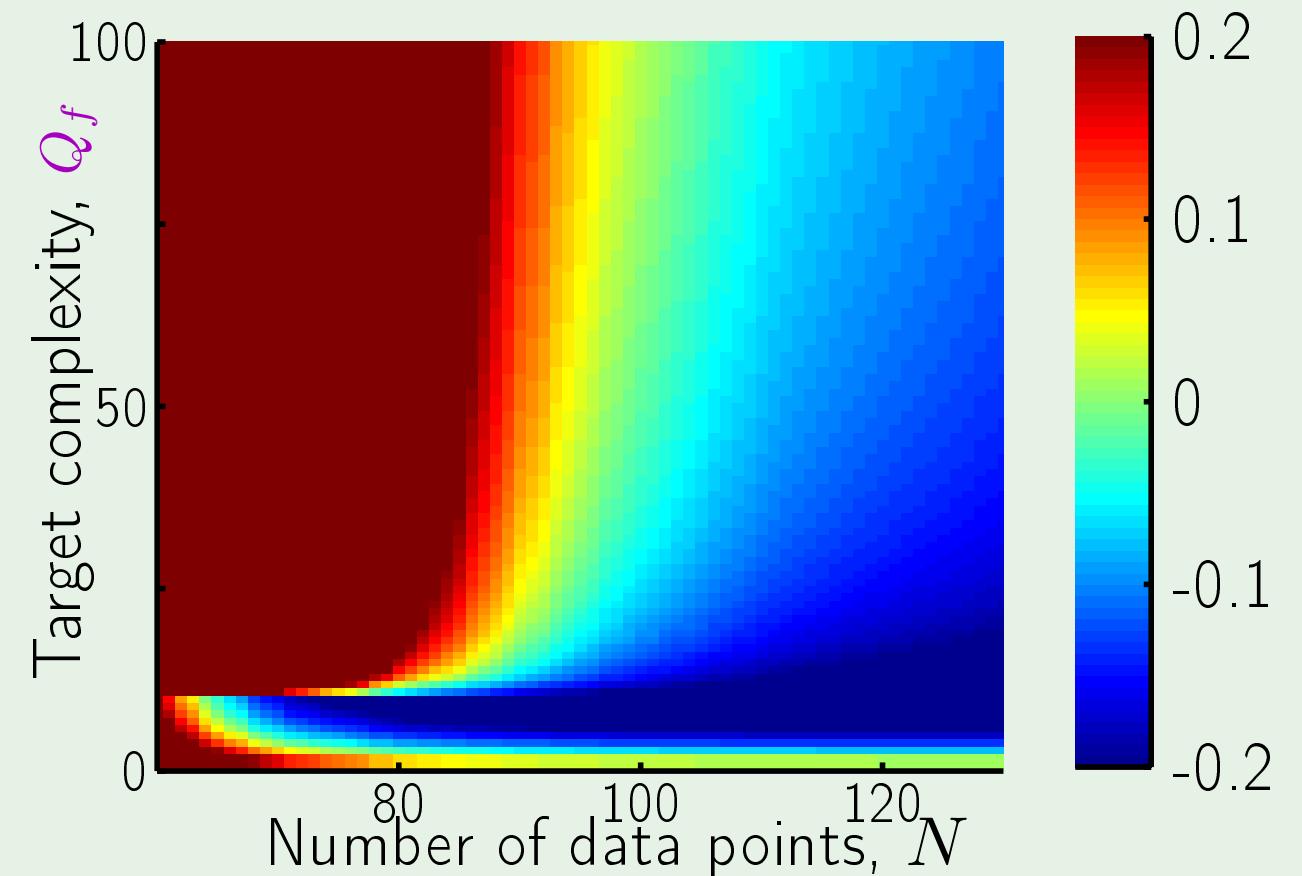
1. depends on \mathcal{H}
2. fixed for a given \mathbf{x}



Impact on overfitting

Deterministic noise and Q_f

Finite N : \mathcal{H} tries to fit the noise



how much overfit

Noise and bias-variance

Recall the decomposition:

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

What if f is a noisy target?

$$y = f(\mathbf{x}) + \epsilon(\mathbf{x}) \quad \mathbb{E} [\epsilon(\mathbf{x})] = 0$$

A noise term

$$\begin{aligned}\mathbb{E}_{\mathcal{D}, \epsilon} \left[(g^{(\mathcal{D})}(\mathbf{x}) - y)^2 \right] &= \mathbb{E}_{\mathcal{D}, \epsilon} \left[(g^{(\mathcal{D})}(\mathbf{x}) - f(\mathbf{x}) - \epsilon(\mathbf{x}))^2 \right] \\ &= \mathbb{E}_{\mathcal{D}, \epsilon} \left[(g^{(\mathcal{D})}(\mathbf{x}) - \bar{g}(\mathbf{x}) + \bar{g}(\mathbf{x}) - f(\mathbf{x}) - \epsilon(\mathbf{x}))^2 \right] \\ &= \mathbb{E}_{\mathcal{D}, \epsilon} \left[(g^{(\mathcal{D})}(\mathbf{x}) - \bar{g}(\mathbf{x}))^2 + (\bar{g}(\mathbf{x}) - f(\mathbf{x}))^2 + (\epsilon(\mathbf{x}))^2 \right. \\ &\quad \left. + \text{ cross terms } \right]\end{aligned}$$

Actually, two noise terms

$$\underbrace{\mathbb{E}_{\mathcal{D}, \mathbf{x}} \left[(g^{(\mathcal{D})}(\mathbf{x}) - \bar{g}(\mathbf{x}))^2 \right]}_{\text{var}} + \underbrace{\mathbb{E}_{\mathbf{x}} \left[(\bar{g}(\mathbf{x}) - f(\mathbf{x}))^2 \right]}_{\text{bias}} + \underbrace{\mathbb{E}_{\epsilon, \mathbf{x}} \left[(\epsilon(\mathbf{x}))^2 \right]}_{\sigma^2}$$

↑ ↑

deterministic noise stochastic noise

Outline

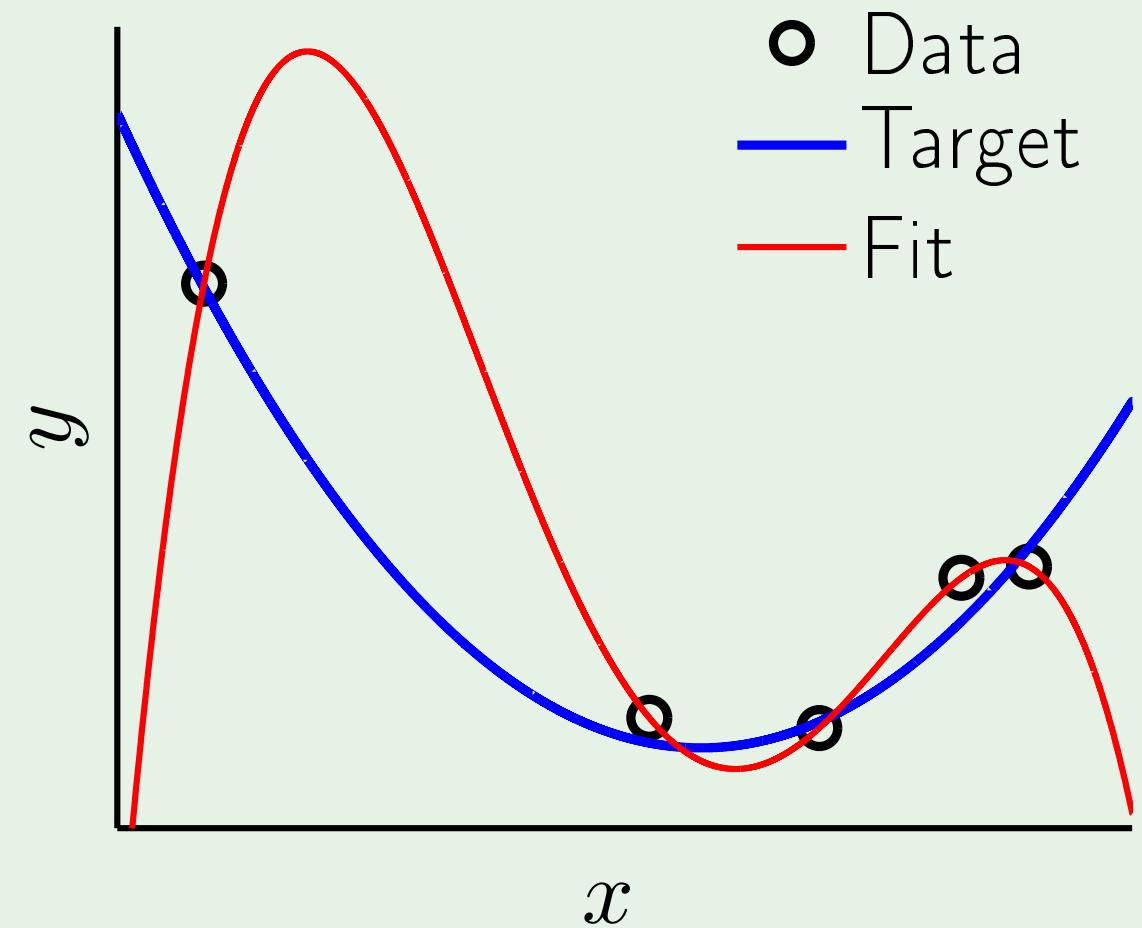
- What is overfitting?
- The role of noise
- Deterministic noise
- Dealing with overfitting

Two cures

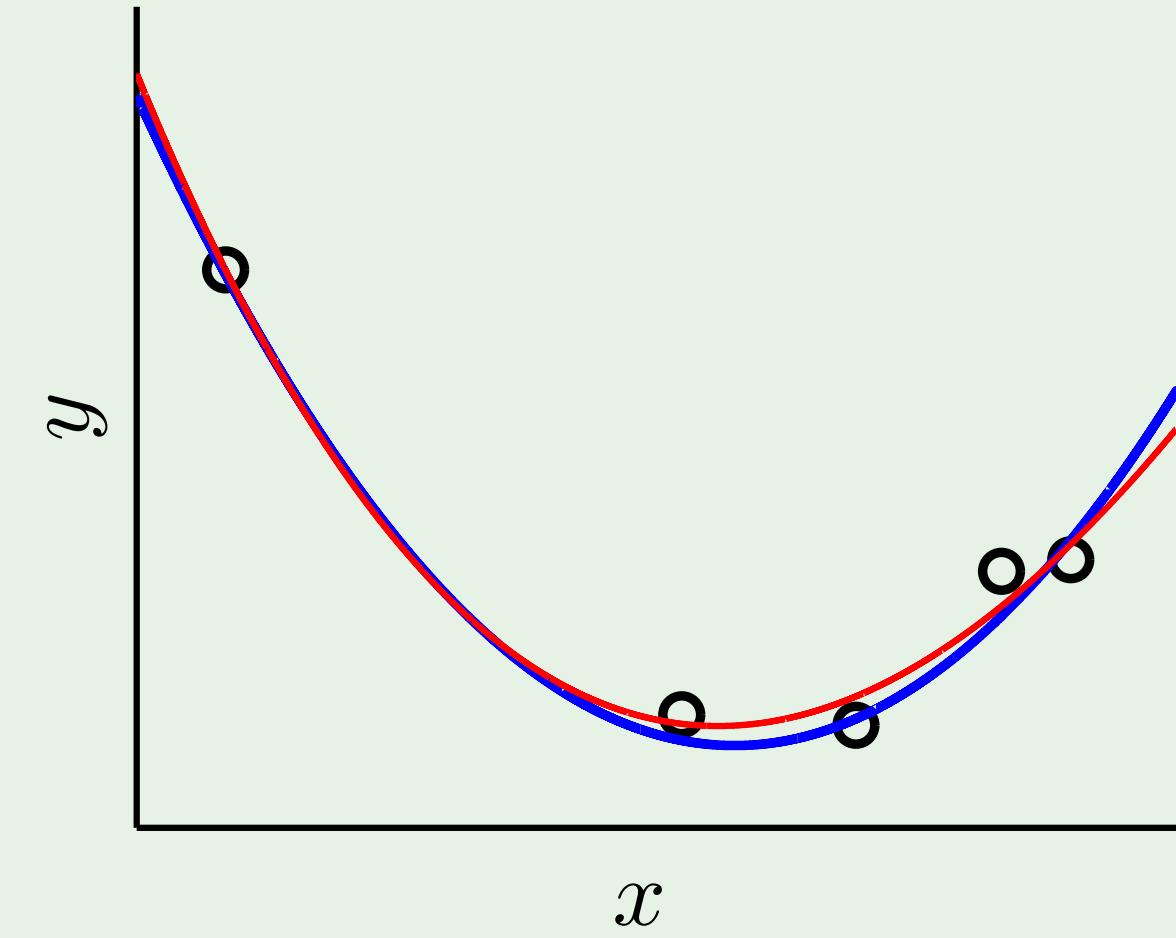
Regularization: Putting the brakes

Validation: Checking the bottom line

Putting the brakes



free fit

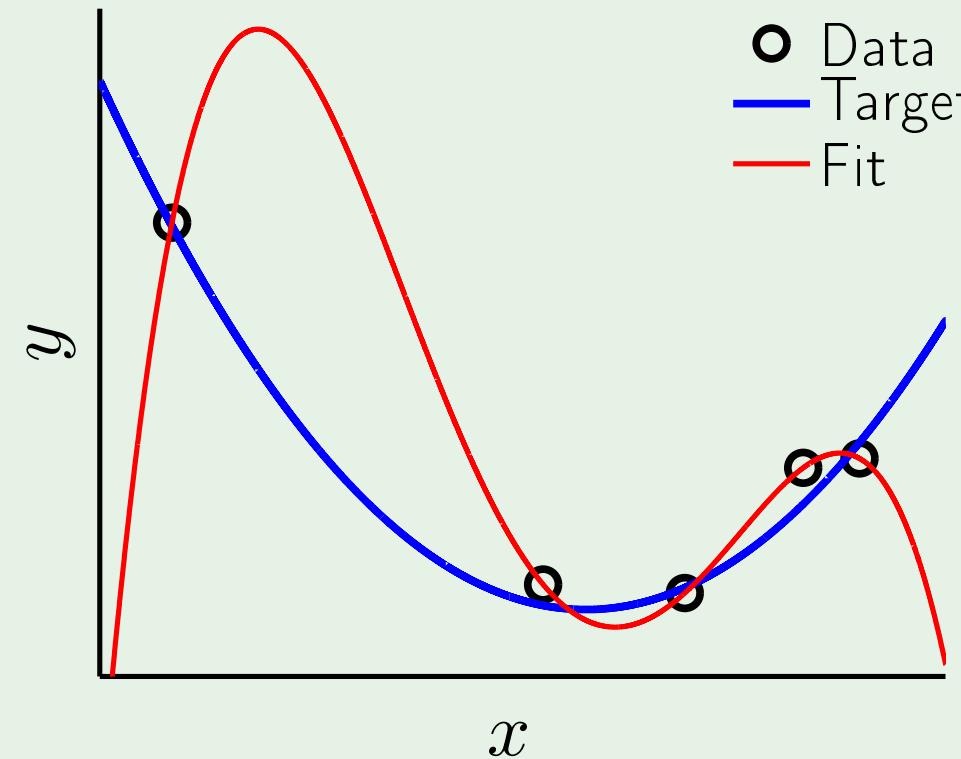


restrained fit

Review of Lecture 11

- Overfitting

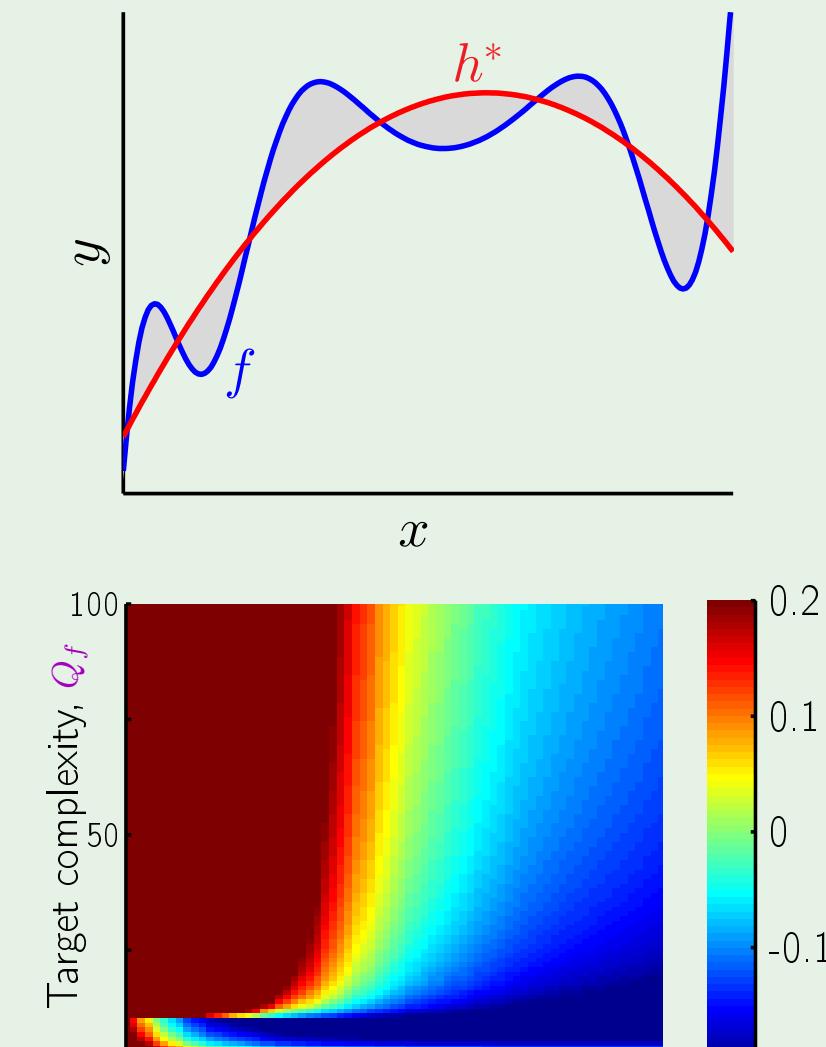
Fitting the data more than is warranted



VC allows it; doesn't predict it

Fitting the noise, stochastic/deterministic

- Deterministic noise



Learning From Data

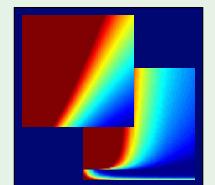
Yaser S. Abu-Mostafa
California Institute of Technology

Lecture 12: Regularization



Sponsored by Caltech's Provost Office, E&AS Division, and IST

• Thursday, May 10, 2012



Outline

- Regularization - informal
- Regularization - formal
- Weight decay
- Choosing a regularizer

Two approaches to regularization

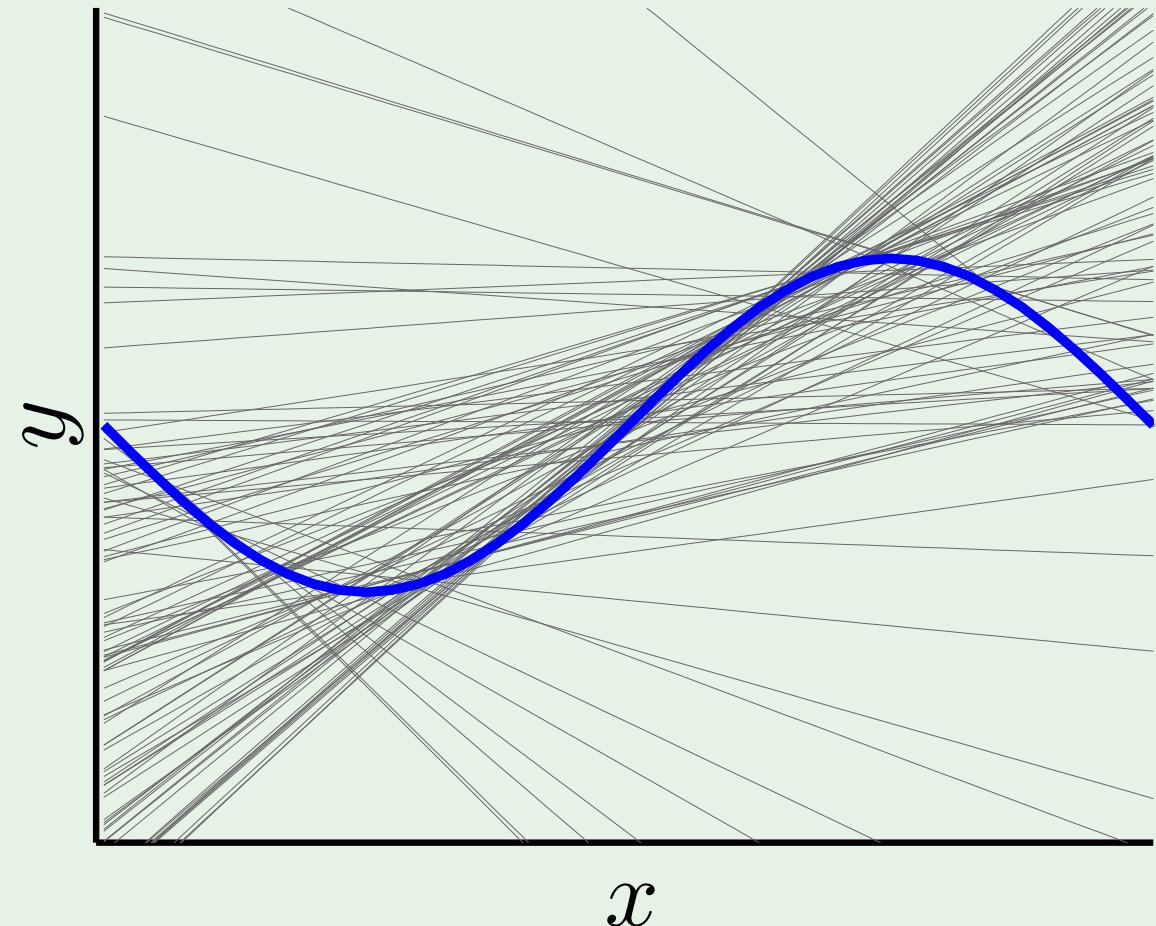
Mathematical:

Ill-posed problems in function approximation

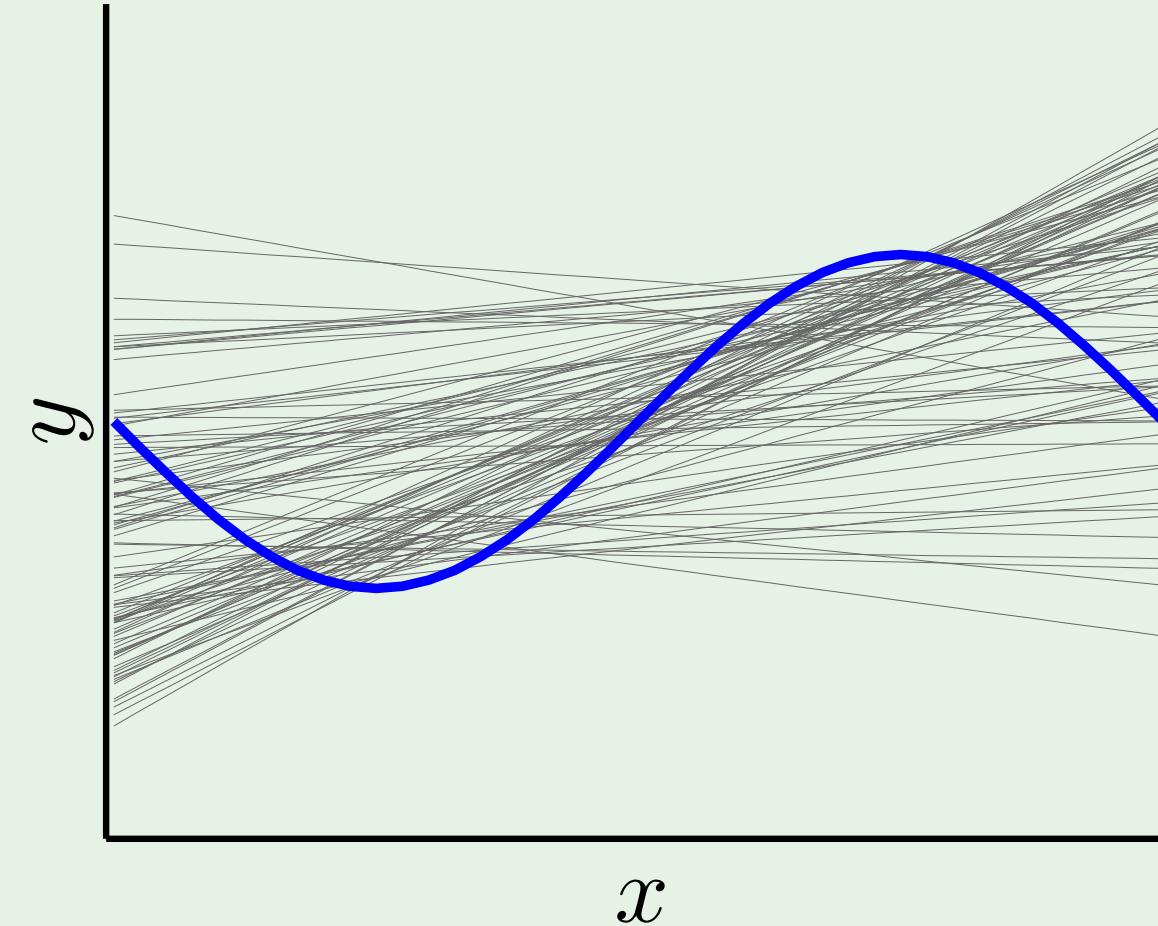
Heuristic:

Handicapping the minimization of E_{in}

A familiar example



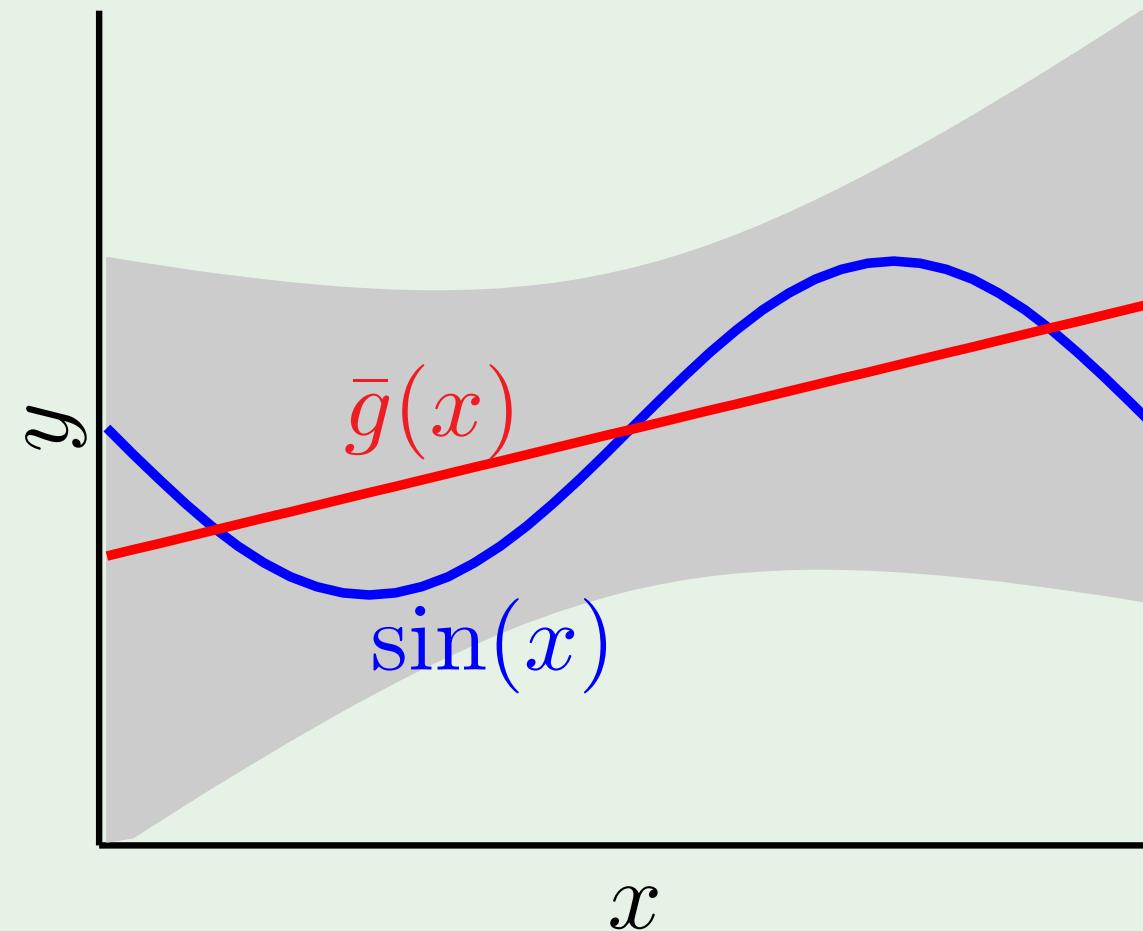
without regularization



with regularization

and the winner is ...

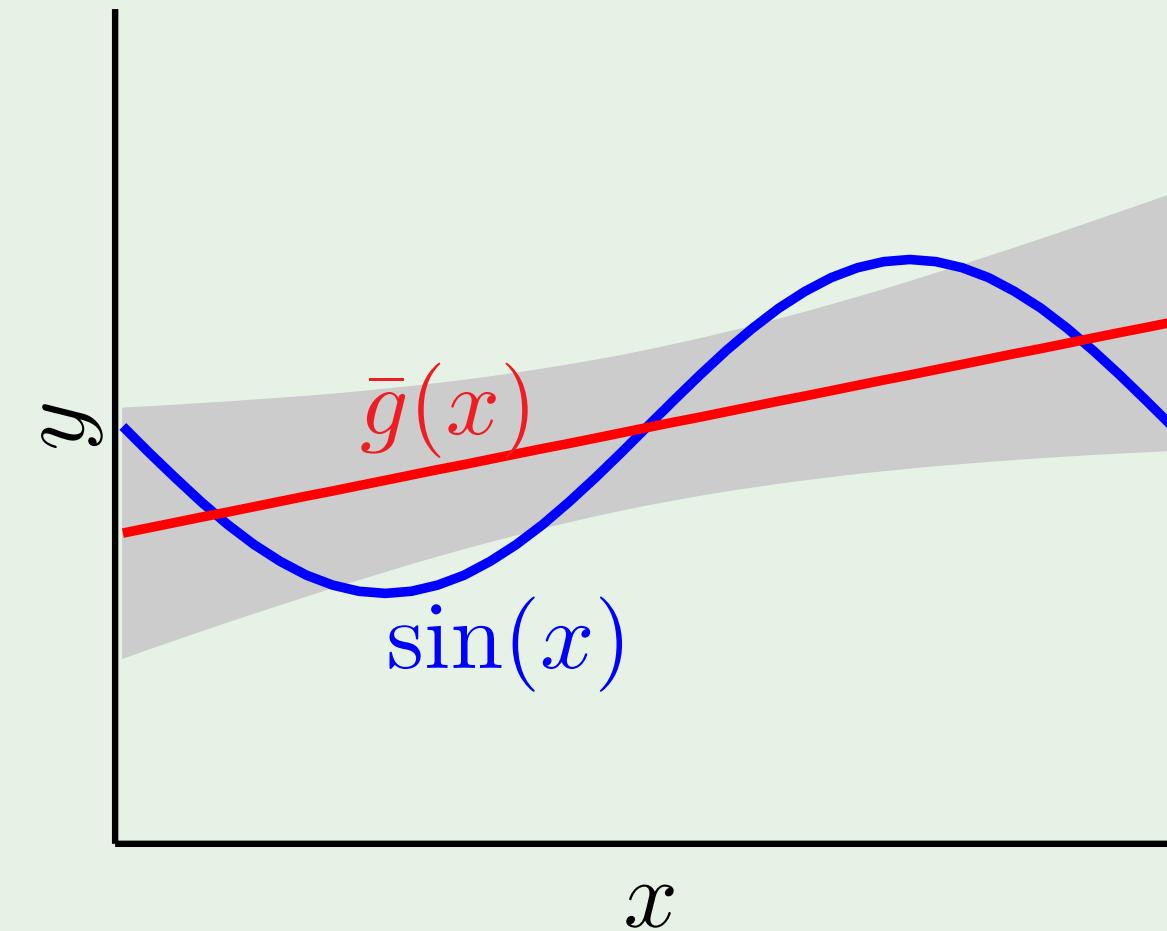
without regularization



bias = **0.21**

var = **1.69**

with regularization



bias = **0.23**

var = **0.33**

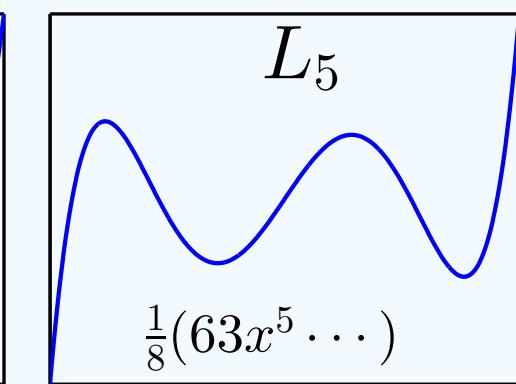
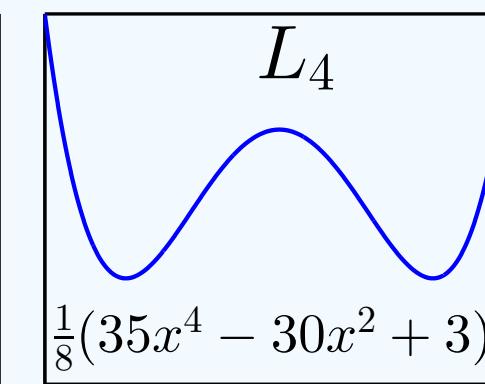
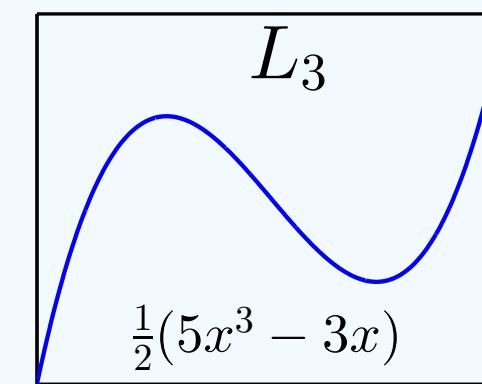
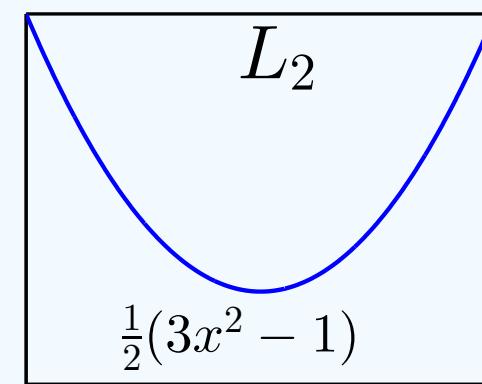
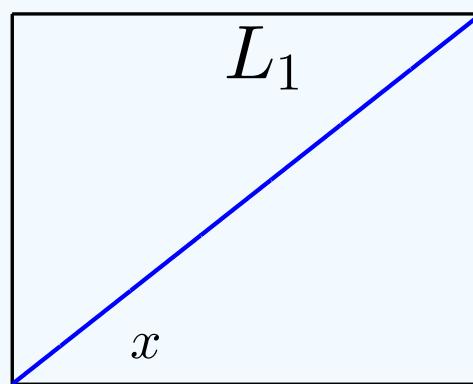
The polynomial model

\mathcal{H}_Q : polynomials of order Q

linear regression in \mathcal{Z} space

$$\mathbf{z} = \begin{bmatrix} 1 \\ L_1(x) \\ \vdots \\ L_Q(x) \end{bmatrix} \quad \mathcal{H}_Q = \left\{ \sum_{q=0}^Q \mathbf{w}_q L_q(x) \right\}$$

Legendre polynomials:



Unconstrained solution

Given $(x_1, y_1), \dots, (x_N, y_n)$ \longrightarrow $(\mathbf{z}_1, y_1), \dots, (\mathbf{z}_N, y_n)$

$$\text{Minimize } E_{\text{in}}(\mathbf{w}) = \frac{1}{N} \sum_{n=1}^N (\mathbf{w}^\top \mathbf{z}_n - y_n)^2$$

$$\text{Minimize } \frac{1}{N} (\mathbf{Z}\mathbf{w} - \mathbf{y})^\top (\mathbf{Z}\mathbf{w} - \mathbf{y})$$

$$\mathbf{w}_{\text{lin}} = (\mathbf{Z}^\top \mathbf{Z})^{-1} \mathbf{Z}^\top \mathbf{y}$$

Constraining the weights

Hard constraint: \mathcal{H}_2 is constrained version of \mathcal{H}_{10} with $w_q = 0$ for $q > 2$

Softer version:
$$\sum_{q=0}^Q w_q^2 \leq C$$
 “soft-order” constraint

$$\text{Minimize } \frac{1}{N} (\mathbf{Z}\mathbf{w} - \mathbf{y})^\top(\mathbf{Z}\mathbf{w} - \mathbf{y})$$

$$\text{subject to: } \mathbf{w}^\top \mathbf{w} \leq C$$

Solution: \mathbf{w}_{reg} instead of \mathbf{w}_{lin}

Solving for \mathbf{w}_{reg}

Minimize $E_{\text{in}}(\mathbf{w}) = \frac{1}{N} (\mathbf{Z}\mathbf{w} - \mathbf{y})^\top (\mathbf{Z}\mathbf{w} - \mathbf{y})$

subject to: $\mathbf{w}^\top \mathbf{w} \leq C$

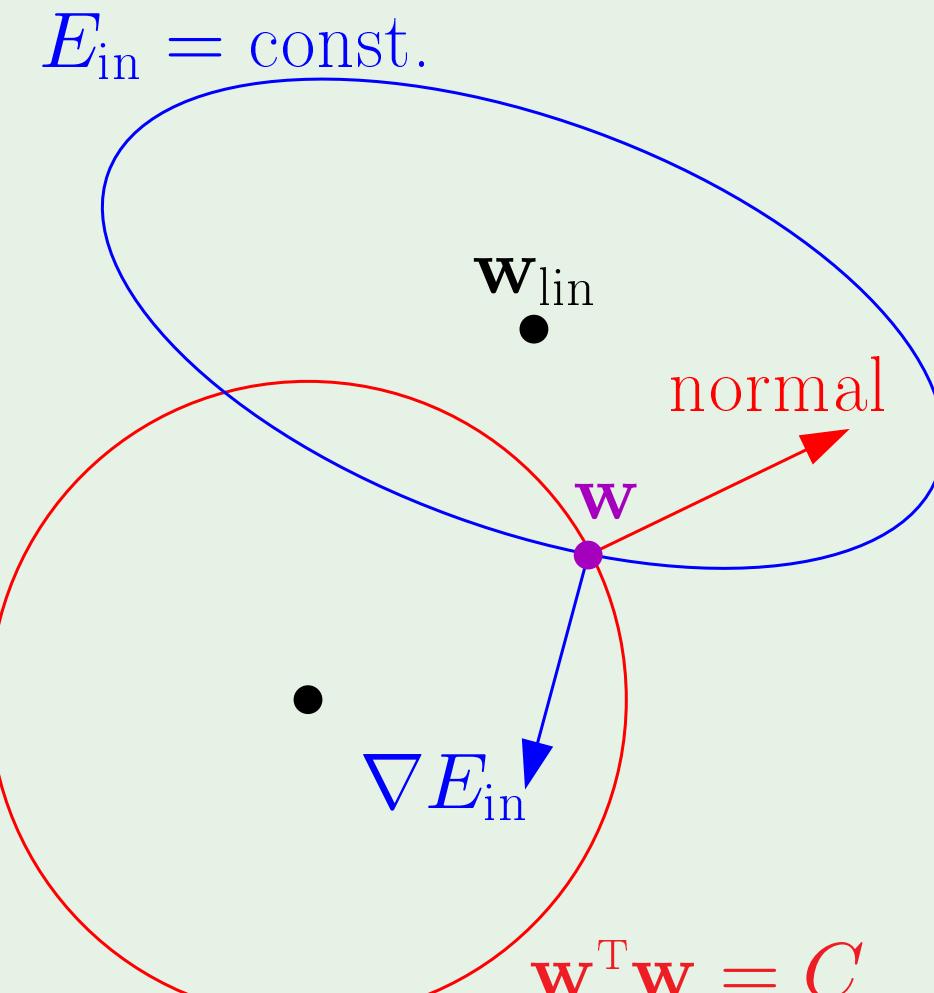
$$\nabla E_{\text{in}}(\mathbf{w}_{\text{reg}}) \propto -\mathbf{w}_{\text{reg}}$$

$$= -2\frac{\lambda}{N}\mathbf{w}_{\text{reg}}$$

$$\nabla E_{\text{in}}(\mathbf{w}_{\text{reg}}) + 2\frac{\lambda}{N}\mathbf{w}_{\text{reg}} = 0$$

Minimize $E_{\text{in}}(\mathbf{w}) + \frac{\lambda}{N}\mathbf{w}^\top \mathbf{w}$

$C \uparrow$	$\lambda \downarrow$
--------------	----------------------



Augmented error

$$\text{Minimizing} \quad E_{\text{aug}}(\mathbf{w}) = E_{\text{in}}(\mathbf{w}) + \frac{\lambda}{N} \mathbf{w}^\top \mathbf{w}$$

$$= \frac{1}{N} (\mathbf{Z}\mathbf{w} - \mathbf{y})^\top (\mathbf{Z}\mathbf{w} - \mathbf{y}) + \frac{\lambda}{N} \mathbf{w}^\top \mathbf{w} \quad \text{unconditionally}$$

— solves —

Minimizing $E_{\text{in}}(\mathbf{w}) = \frac{1}{N} (\mathbf{Z}\mathbf{w} - \mathbf{y})^\top(\mathbf{Z}\mathbf{w} - \mathbf{y})$
 subject to: $\mathbf{w}^\top \mathbf{w} \leq C$ \leftarrow VC formulation

The solution

Minimize

$$\begin{aligned} E_{\text{aug}}(\mathbf{w}) &= E_{\text{in}}(\mathbf{w}) + \frac{\lambda}{N} \mathbf{w}^T \mathbf{w} \\ &= \frac{1}{N} \left((\mathbf{Z}\mathbf{w} - \mathbf{y})^T (\mathbf{Z}\mathbf{w} - \mathbf{y}) + \lambda \mathbf{w}^T \mathbf{w} \right) \end{aligned}$$

$$\nabla E_{\text{aug}}(\mathbf{w}) = \mathbf{0} \implies \mathbf{Z}^T (\mathbf{Z}\mathbf{w} - \mathbf{y}) + \lambda \mathbf{w} = \mathbf{0}$$

$$\mathbf{w}_{\text{reg}} = (\mathbf{Z}^T \mathbf{Z} + \lambda \mathbf{I})^{-1} \mathbf{Z}^T \mathbf{y}$$

(with regularization)

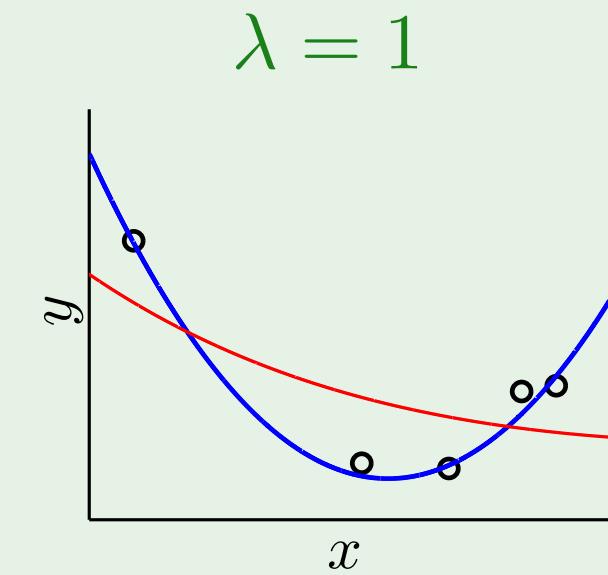
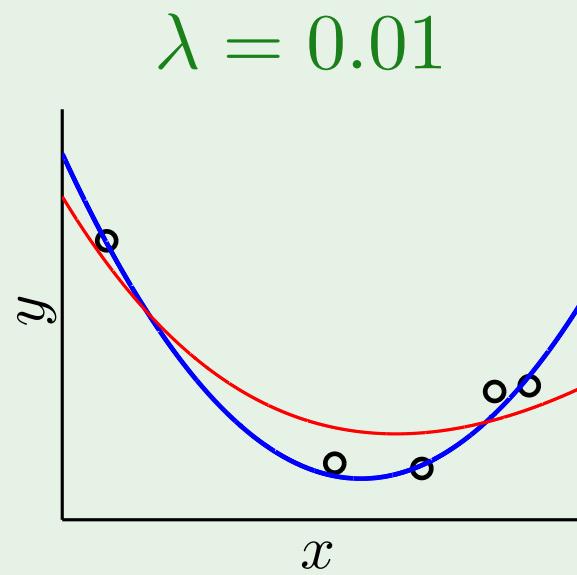
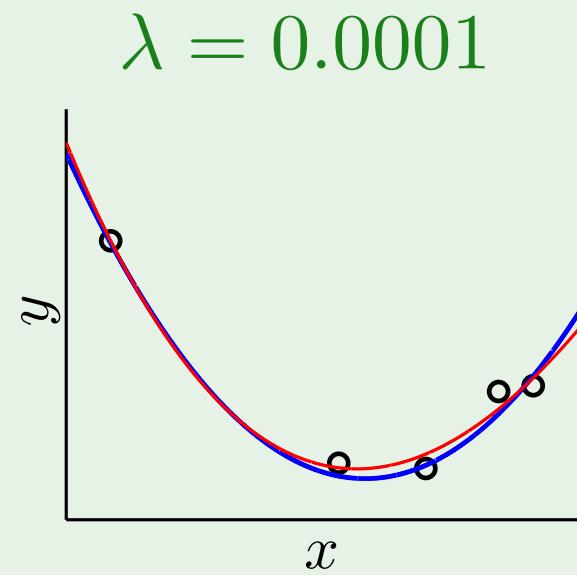
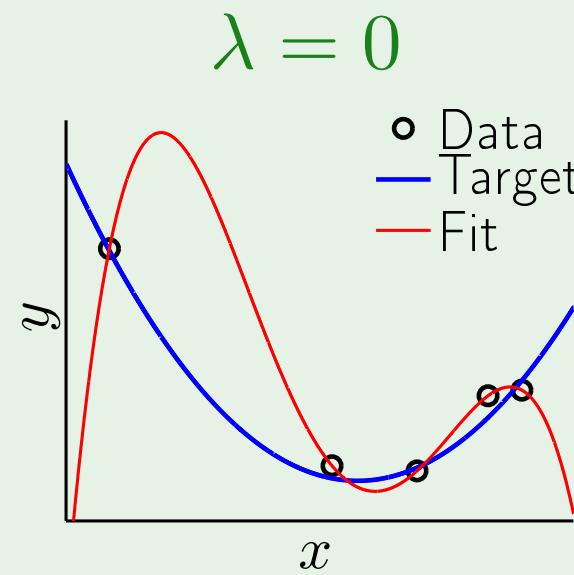
as opposed to

$$\mathbf{w}_{\text{lin}} = (\mathbf{Z}^T \mathbf{Z})^{-1} \mathbf{Z}^T \mathbf{y}$$

(without regularization)

The result

Minimizing $E_{\text{in}}(\mathbf{w}) + \frac{\lambda}{N} \mathbf{w}^\top \mathbf{w}$ for different λ 's:



overfitting



underfitting

Weight ‘decay’

Minimizing $E_{\text{in}}(\mathbf{w}) + \frac{\lambda}{N} \mathbf{w}^\top \mathbf{w}$ is called weight *decay*. Why?

Gradient descent:

$$\begin{aligned}\mathbf{w}(t+1) &= \mathbf{w}(t) - \eta \nabla E_{\text{in}}(\mathbf{w}(t)) - 2\eta \frac{\lambda}{N} \mathbf{w}(t) \\ &= \mathbf{w}(t) \left(1 - 2\eta \frac{\lambda}{N}\right) - \eta \nabla E_{\text{in}}(\mathbf{w}(t))\end{aligned}$$

Applies in neural networks:

$$\mathbf{w}^\top \mathbf{w} = \sum_{l=1}^L \sum_{i=0}^{d^{(l-1)}} \sum_{j=1}^{d^{(l)}} \left(w_{ij}^{(l)}\right)^2$$

Variations of weight decay

Emphasis of certain weights:

$$\sum_{q=0}^Q \gamma_q w_q^2$$

Examples:

$\gamma_q = 2^q \Rightarrow$ low-order fit

$\gamma_q = 2^{-q} \Rightarrow$ high-order fit

Neural networks: different layers get different γ 's

Tikhonov regularizer: $\mathbf{w}^\top \Gamma^\top \Gamma \mathbf{w}$

Even weight growth!

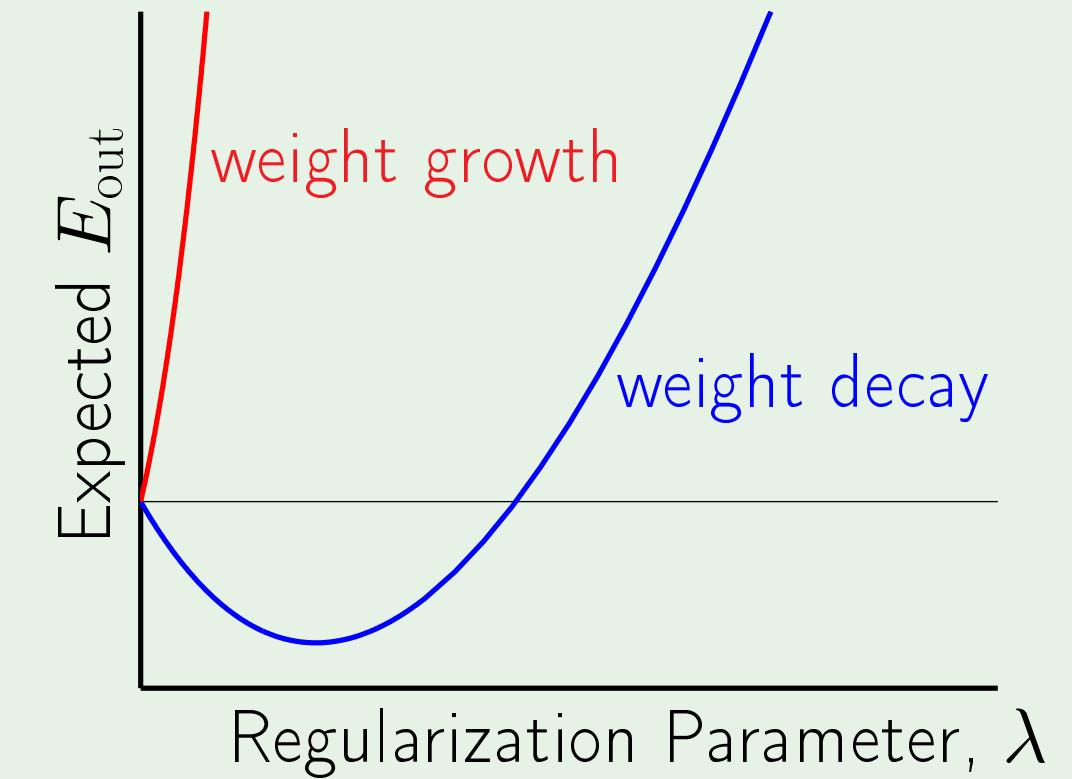
We ‘constrain’ the weights to be large - bad!

Practical rule:

stochastic noise is ‘high-frequency’

deterministic noise is also non-smooth

⇒ constrain learning towards smoother hypotheses

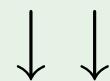


General form of augmented error

Calling the regularizer $\Omega = \Omega(h)$, we minimize

$$E_{\text{aug}}(h) = E_{\text{in}}(h) + \frac{\lambda}{N} \Omega(h)$$

Rings a bell?



$$E_{\text{out}}(h) \leq E_{\text{in}}(h) + \Omega(\mathcal{H})$$

E_{aug} is better than E_{in} as a proxy for E_{out}

Outline

- Regularization - informal
- Regularization - formal
- Weight decay
- Choosing a regularizer

The perfect regularizer Ω

Constraint in the ‘direction’ of the target function (going in circles ☺)

Guiding principle:

Direction of **smoother** or “simpler”

Chose a bad Ω ?

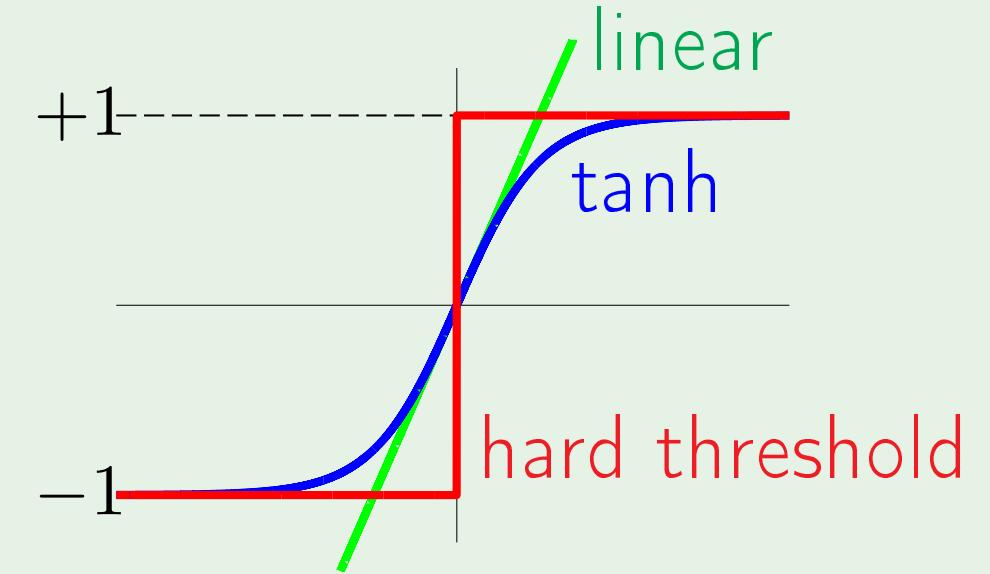
We still have λ !

Neural-network regularizers

Weight decay: From linear to logical

Weight elimination:

Fewer weights \implies smaller VC dimension



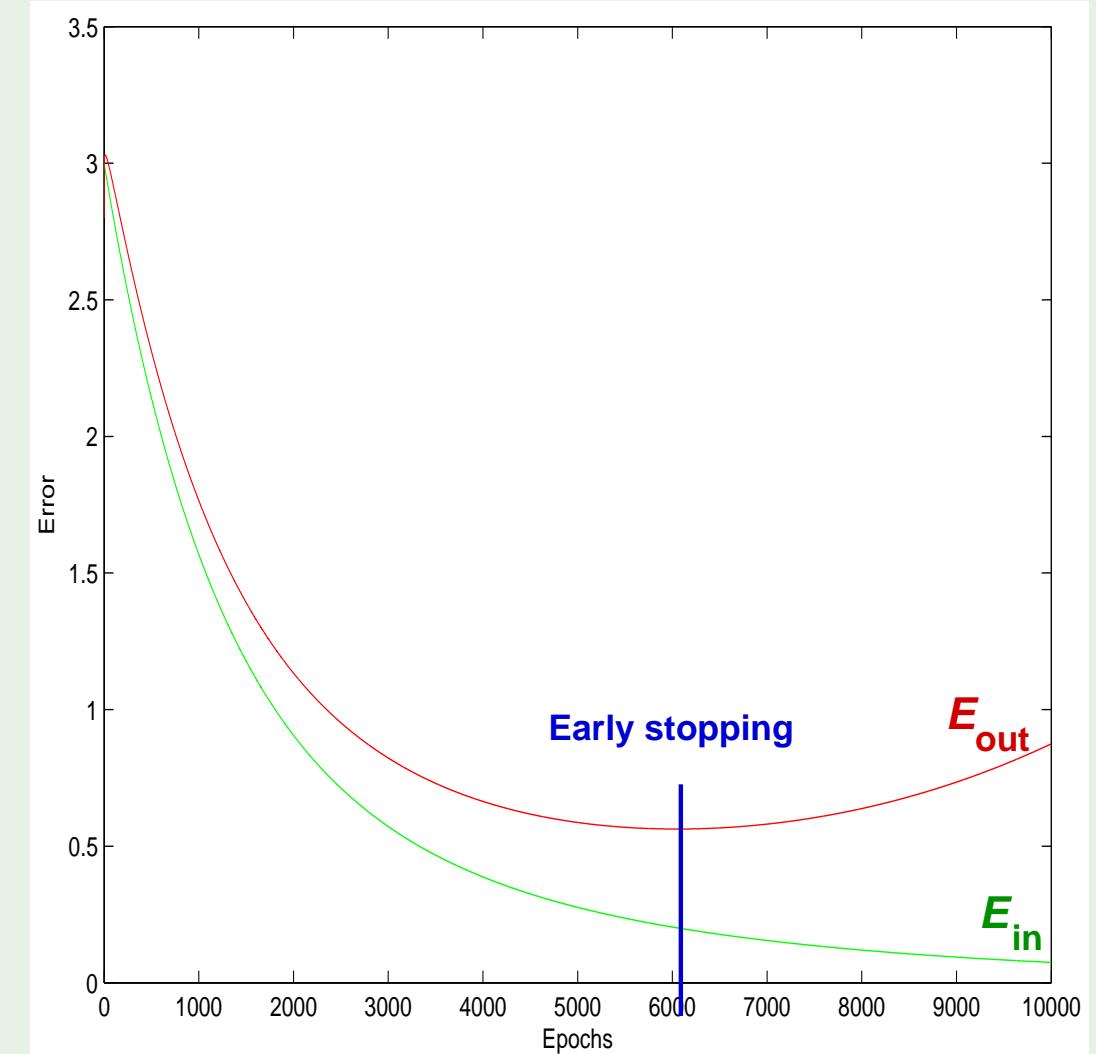
Soft weight elimination:

$$\Omega(\mathbf{w}) = \sum_{i,j,l} \frac{\left(w_{ij}^{(l)}\right)^2}{\beta^2 + \left(w_{ij}^{(l)}\right)^2}$$

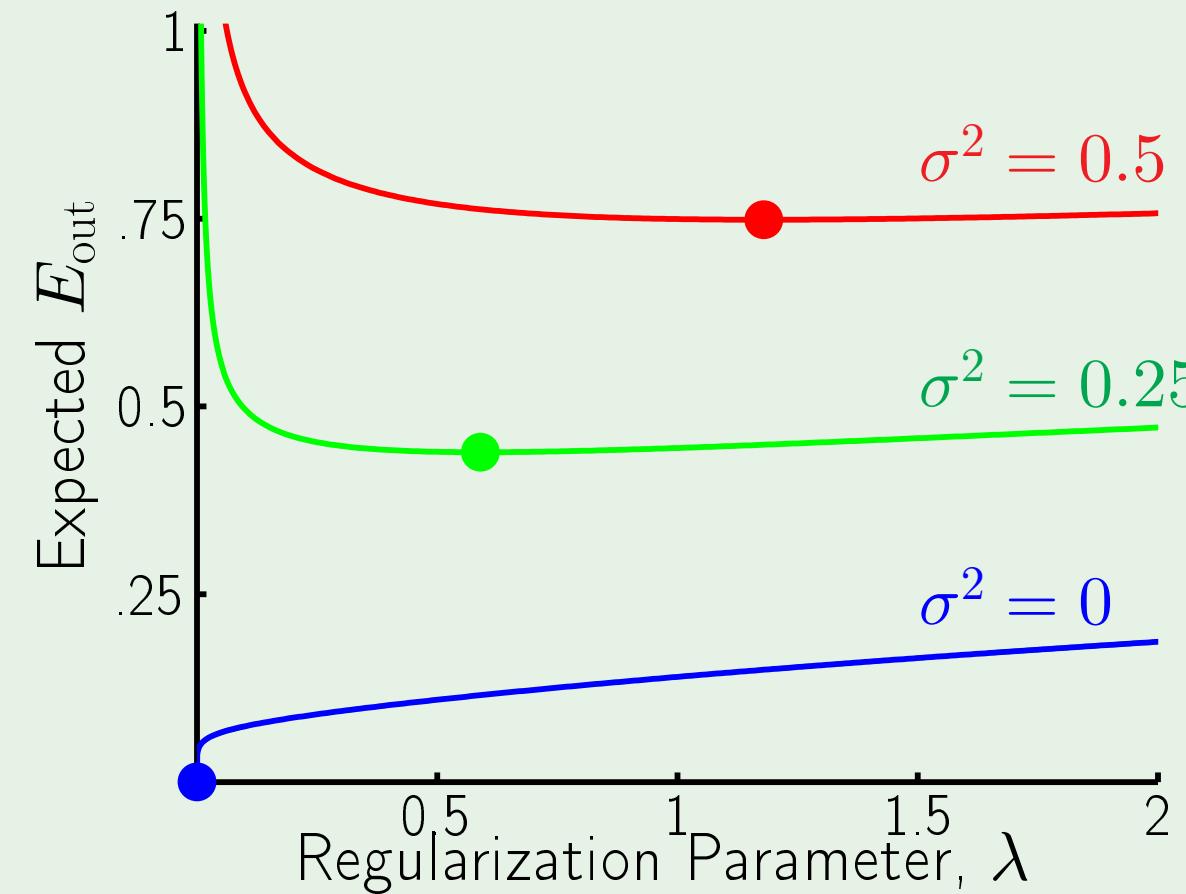
Early stopping as a regularizer

Regularization through the optimizer!

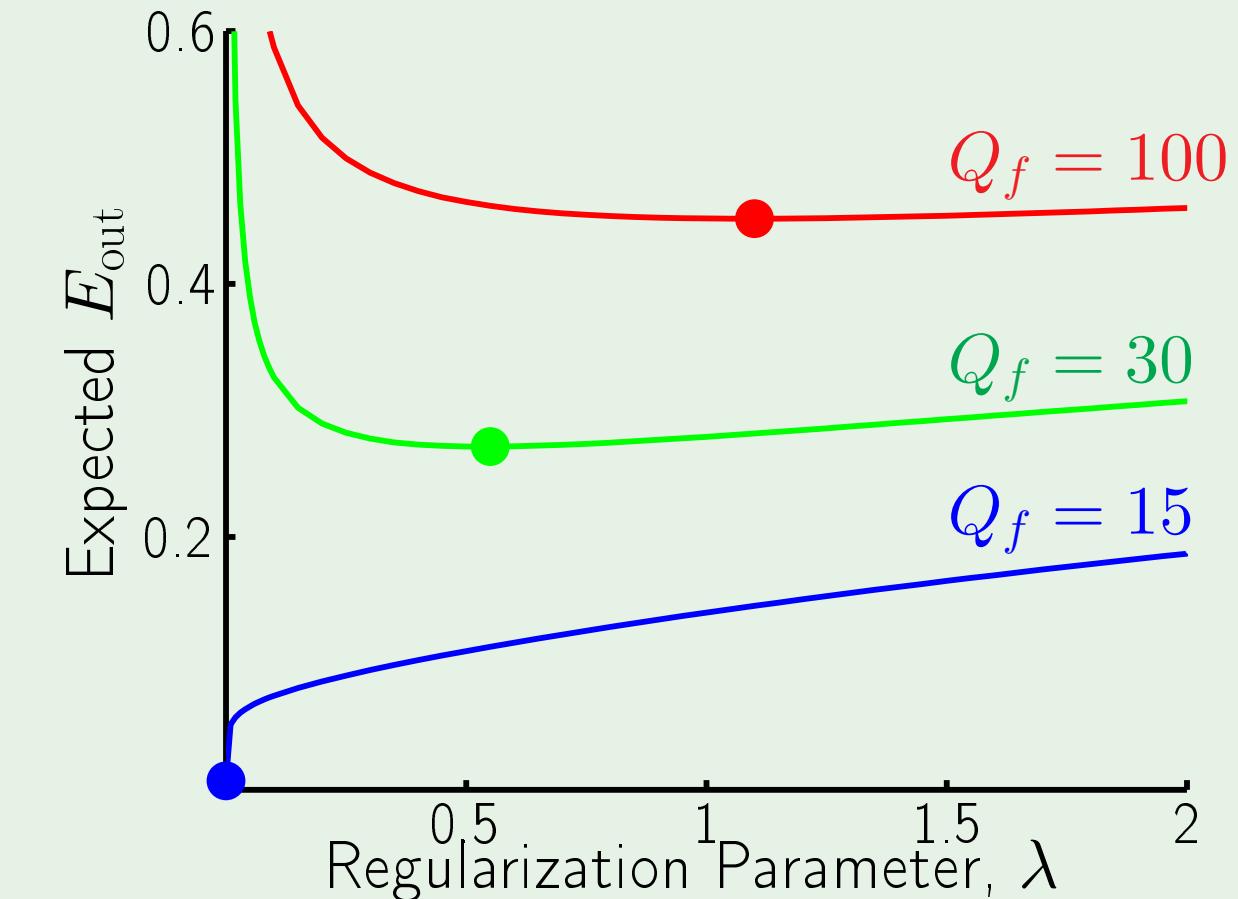
When to stop? **validation**



The optimal λ



Stochastic noise

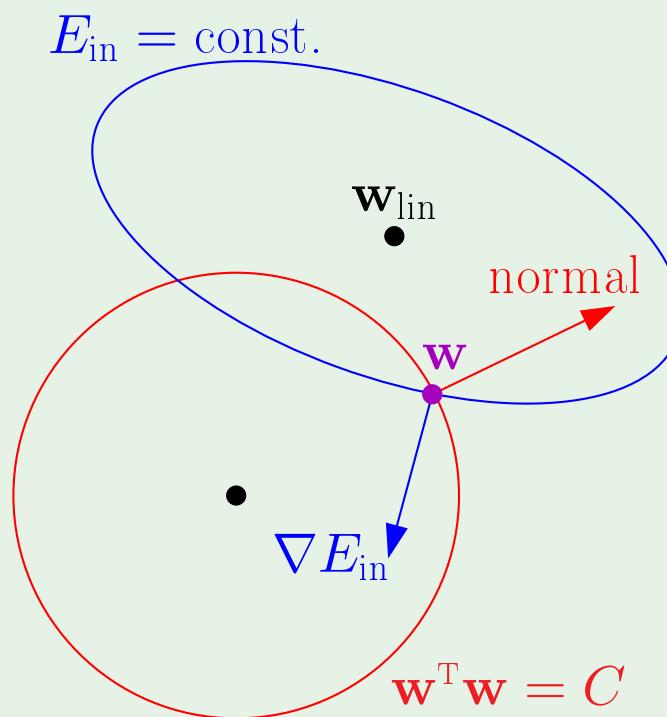


Deterministic noise

Review of Lecture 12

• Regularization

constrained \longrightarrow unconstrained



$$\text{Minimize } E_{\text{aug}}(\mathbf{w}) = E_{\text{in}}(\mathbf{w}) + \frac{\lambda}{N} \mathbf{w}^T \mathbf{w}$$

• Choosing a regularizer

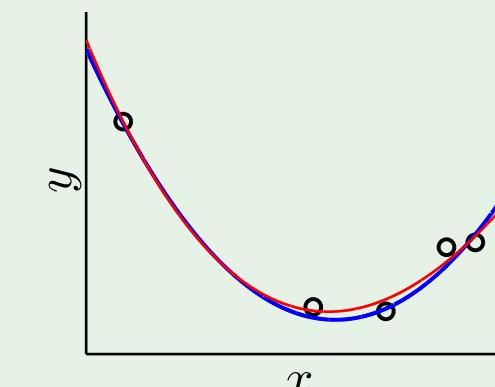
$$E_{\text{aug}}(h) = E_{\text{in}}(h) + \frac{\lambda}{N} \Omega(h)$$

$\Omega(h)$: heuristic \rightarrow smooth, simple h

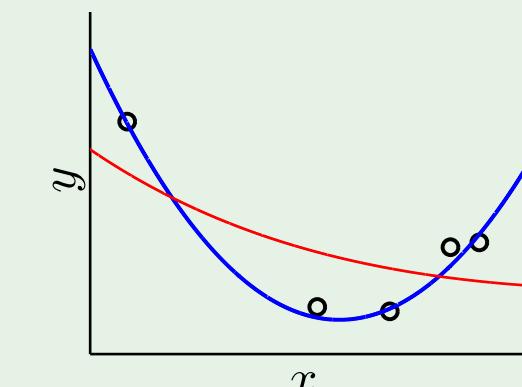
most used: **weight decay**

λ : principled; validation

$$\lambda = 0.0001$$



$$\lambda = 1.0$$



Learning From Data

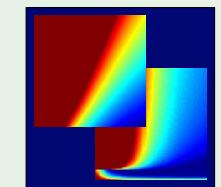
Yaser S. Abu-Mostafa
California Institute of Technology

Lecture 13: Validation



Sponsored by Caltech's Provost Office, E&AS Division, and IST

• Tuesday, May 15, 2012



Outline

- The validation set
- Model selection
- Cross validation

Validation versus regularization

In one form or another, $E_{\text{out}}(h) = E_{\text{in}}(h) + \text{overfit penalty}$

Regularization:

$$E_{\text{out}}(h) = E_{\text{in}}(h) + \underbrace{\text{overfit penalty}}_{\text{regularization estimates this quantity}}$$

Validation:

$$\underbrace{E_{\text{out}}(h)}_{\text{validation estimates this quantity}} = E_{\text{in}}(h) + \text{overfit penalty}$$

Analyzing the estimate

On out-of-sample point (\mathbf{x}, y) , the error is $\mathbf{e}(h(\mathbf{x}), y)$

Squared error: $(h(\mathbf{x}) - y)^2$

Binary error: $\llbracket h(\mathbf{x}) \neq y \rrbracket$

$$\mathbb{E} [\mathbf{e}(h(\mathbf{x}), y)] = E_{\text{out}}(h)$$

$$\text{var} [\mathbf{e}(h(\mathbf{x}), y)] = \sigma^2$$

From a point to a set

On a validation set $(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_K, y_K)$, the error is $E_{\text{val}}(h) = \frac{1}{K} \sum_{k=1}^K \mathbf{e}(h(\mathbf{x}_k), y_k)$

$$\mathbb{E} [E_{\text{val}}(h)] = \frac{1}{K} \sum_{k=1}^K \mathbb{E} [\mathbf{e}(h(\mathbf{x}_k), y_k)] = E_{\text{out}}(h)$$

$$\text{var} [E_{\text{val}}(h)] = \frac{1}{K^2} \sum_{k=1}^K \text{var} [\mathbf{e}(h(\mathbf{x}_k), y_k)] = \frac{\sigma^2}{K}$$

$$E_{\text{val}}(h) = E_{\text{out}}(h) \pm O\left(\frac{1}{\sqrt{K}}\right)$$

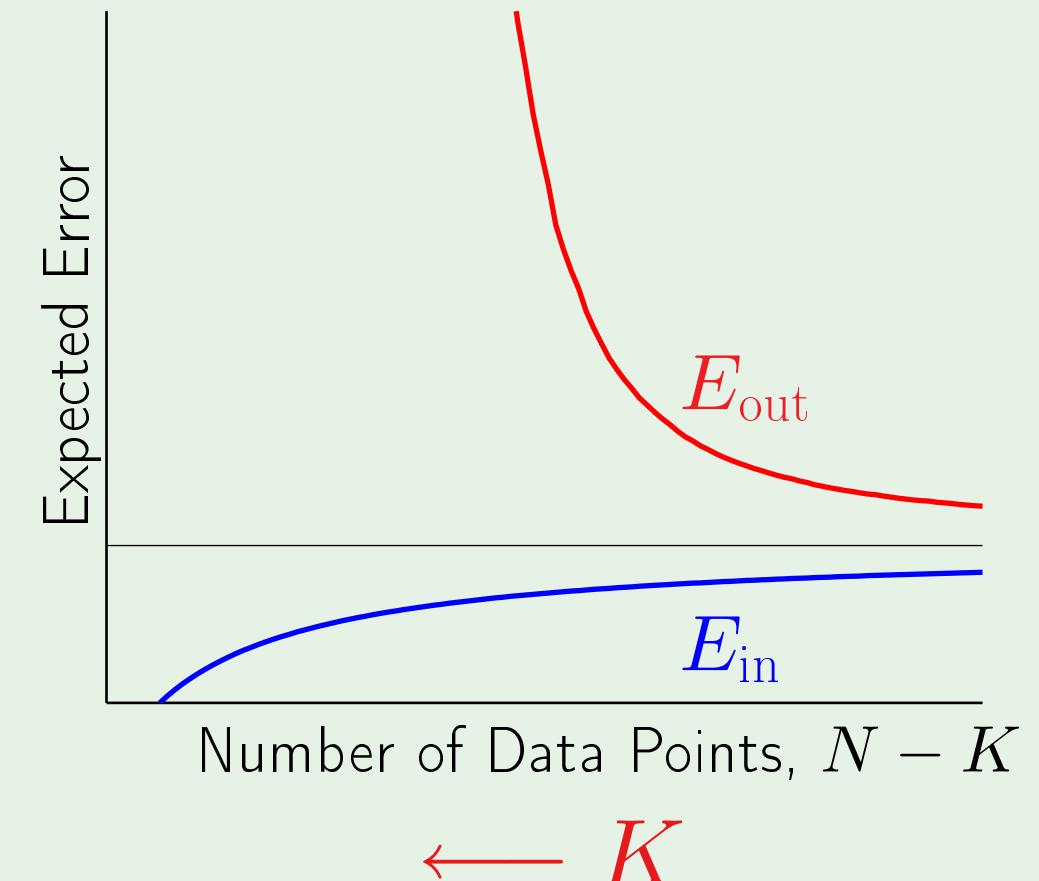
K is taken out of N

Given the data set $\mathcal{D} = (\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)$

$\underbrace{K \text{ points}}_{\mathcal{D}_{\text{val}}} \rightarrow \text{validation}$ $\underbrace{N - K \text{ points}}_{\mathcal{D}_{\text{train}}} \rightarrow \text{training}$

$O\left(\frac{1}{\sqrt{K}}\right)$: Small $K \implies$ bad estimate

Large $K \implies ?$



K is put back into N

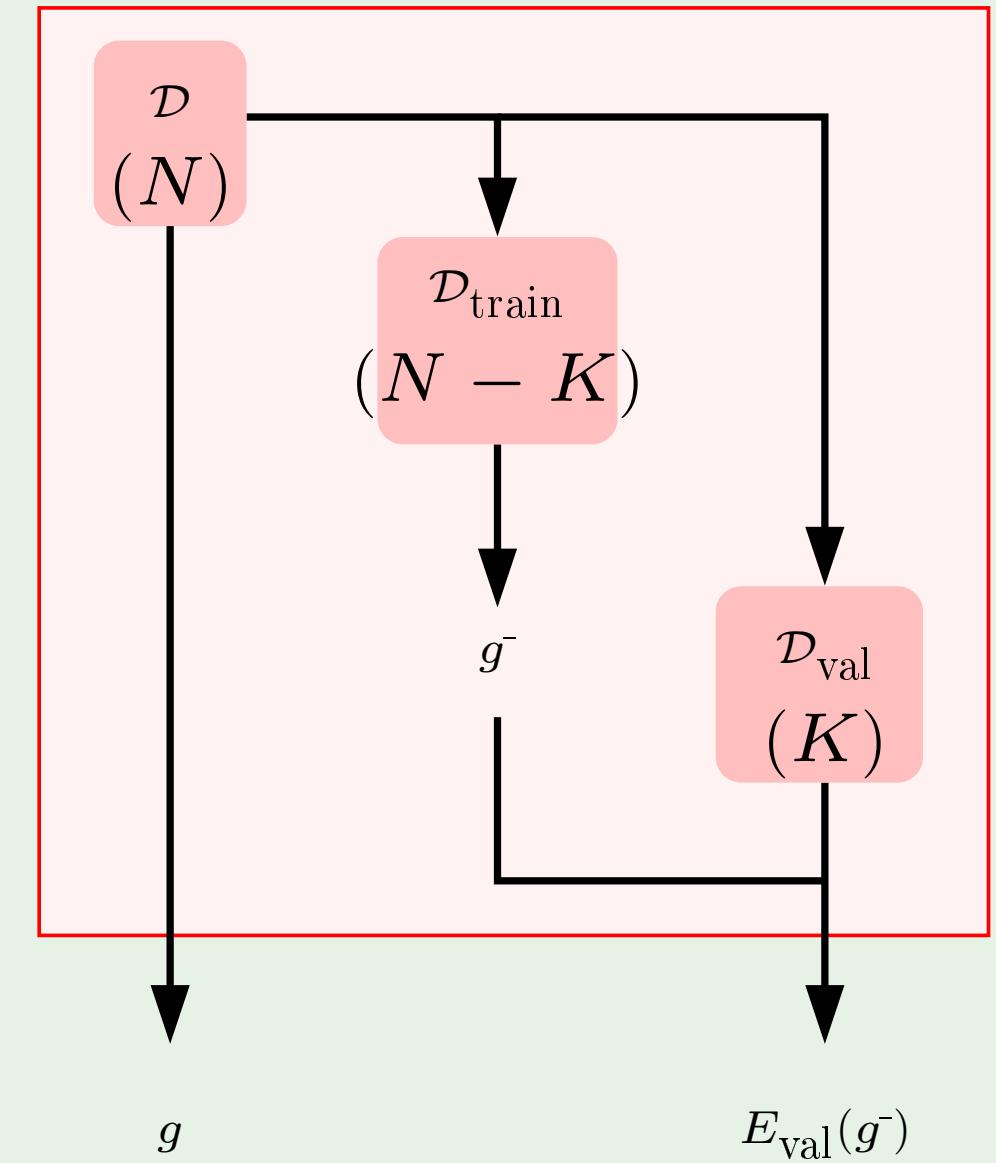
$$\begin{array}{ccc} \mathcal{D} & \longrightarrow & \mathcal{D}_{\text{train}} \cup \mathcal{D}_{\text{val}} \\ \downarrow & & \downarrow \\ N & & N - K & K \end{array}$$

$$\mathcal{D} \implies g \quad \mathcal{D}_{\text{train}} \implies g^-$$

$$E_{\text{val}} = E_{\text{val}}(g^-) \quad \text{Large } K \implies \text{bad estimate!}$$

Rule of Thumb:

$$K = \frac{N}{5}$$



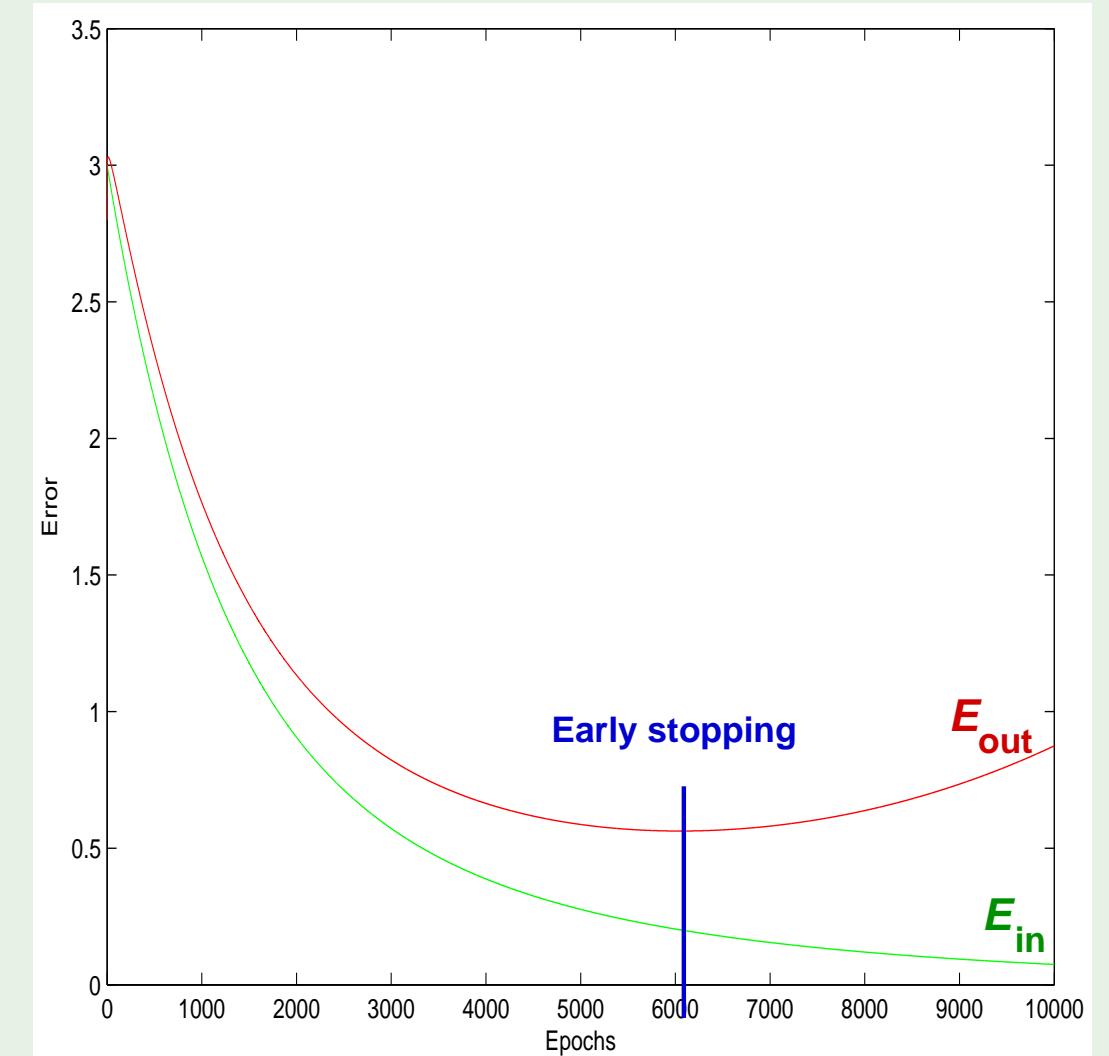
Why ‘validation’

\mathcal{D}_{val} is used to make learning choices

If an estimate of E_{out} affects learning:

the set is no longer a **test** set!

It becomes a **validation** set



What's the difference?

Test set is unbiased; validation set has optimistic bias

Two hypotheses h_1 and h_2 with $E_{\text{out}}(h_1) = E_{\text{out}}(h_2) = 0.5$

Error estimates \mathbf{e}_1 and \mathbf{e}_2 uniform on $[0, 1]$

Pick $h \in \{h_1, h_2\}$ with $\mathbf{e} = \min(\mathbf{e}_1, \mathbf{e}_2)$

$\mathbb{E}(\mathbf{e}) < 0.5$ optimistic bias

Outline

- The validation set
- Model selection
- Cross validation

Using \mathcal{D}_{val} more than once

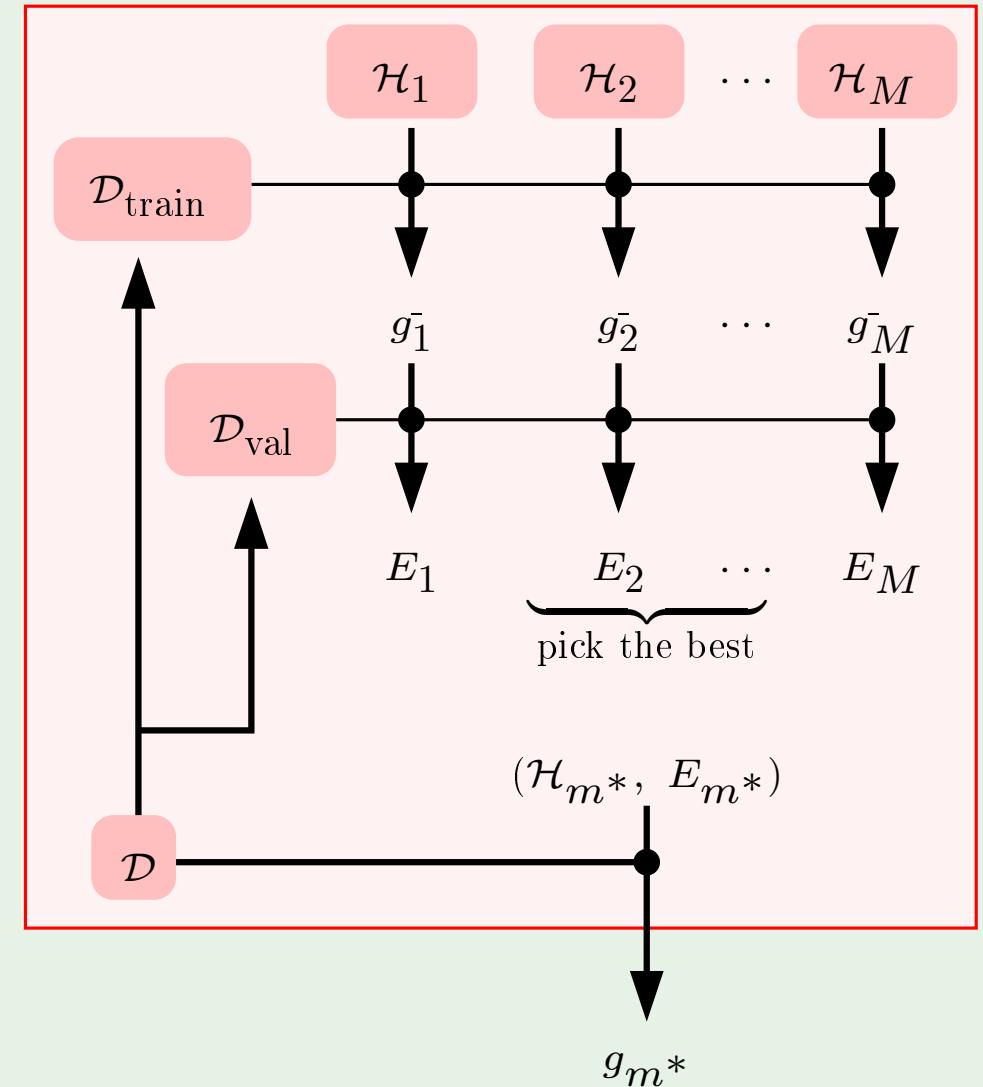
M models $\mathcal{H}_1, \dots, \mathcal{H}_M$

Use $\mathcal{D}_{\text{train}}$ to learn $\bar{g_m}$ for each model

Evaluate $\bar{g_m}$ using \mathcal{D}_{val} :

$$E_m = E_{\text{val}}(\bar{g_m}); \quad m = 1, \dots, M$$

Pick model $m = m^*$ with smallest E_m

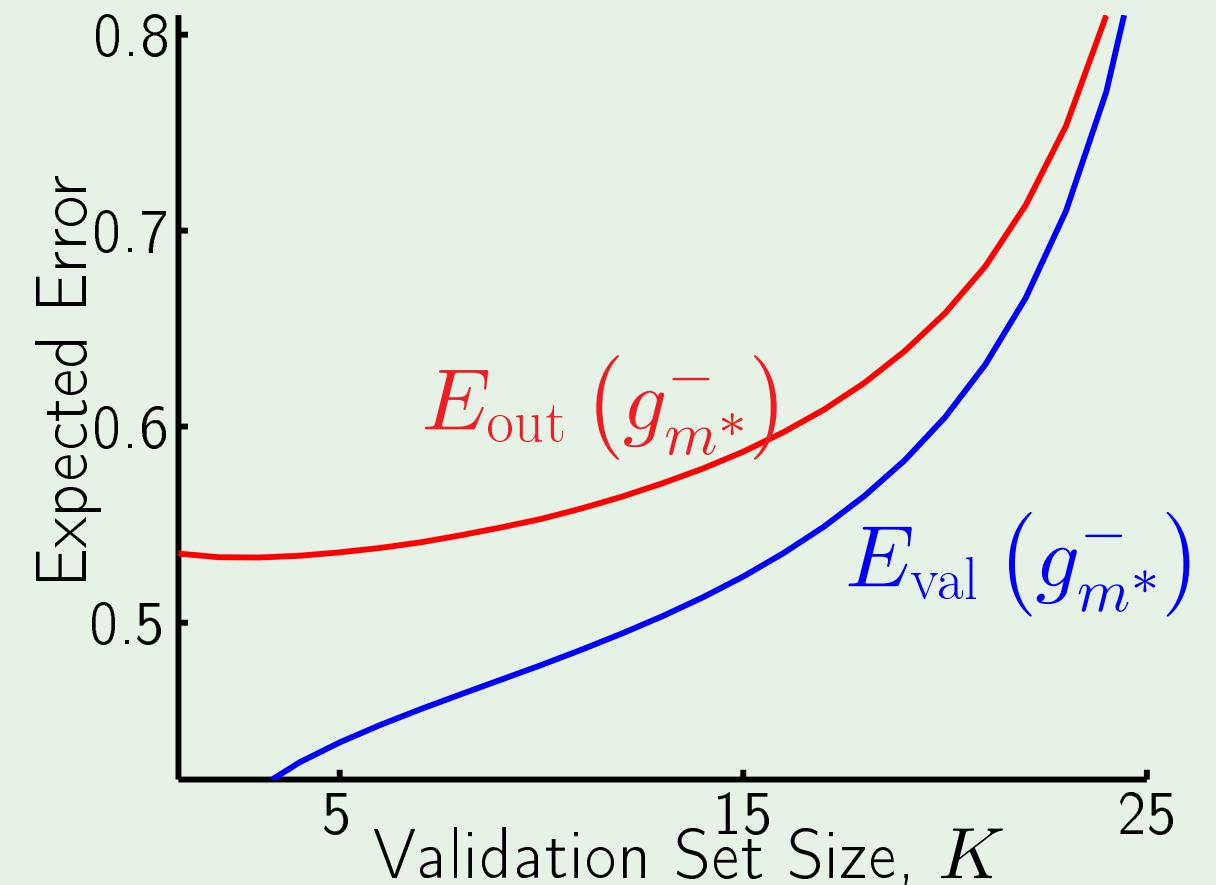


The bias

We selected the model \mathcal{H}_{m^*} using \mathcal{D}_{val}

$E_{\text{val}}(g_{m^*}^-)$ is a biased estimate of $E_{\text{out}}(g_{m^*}^-)$

Illustration: selecting between 2 models



How much bias

For M models: $\mathcal{H}_1, \dots, \mathcal{H}_M$

\mathcal{D}_{val} is used for “training” on the **finalists model**:

$$\mathcal{H}_{\text{val}} = \{g_1^-, g_2^-, \dots, g_M^-\}$$

Back to Hoeffding and VC!

$$E_{\text{out}}(g_{m^*}^-) \leq E_{\text{val}}(g_{m^*}^-) + O\left(\sqrt{\frac{\ln M}{K}}\right)$$

regularization λ early-stopping T

Data contamination

Error estimates: E_{in} , E_{test} , E_{val}

Contamination: Optimistic (deceptive) bias in estimating E_{out}

Training set: totally contaminated

Validation set: slightly contaminated

Test set: totally ‘clean’

Outline

- The validation set
- Model selection
- Cross validation

The dilemma about K

The following chain of reasoning:

$$E_{\text{out}}(g) \approx E_{\text{out}}(g^-) \approx E_{\text{val}}(g^-)$$

(small K) (large K)

highlights the dilemma in selecting K :

Can we have K both small and large? ☺

Leave one out

$N - 1$ points for training, and 1 point for validation!

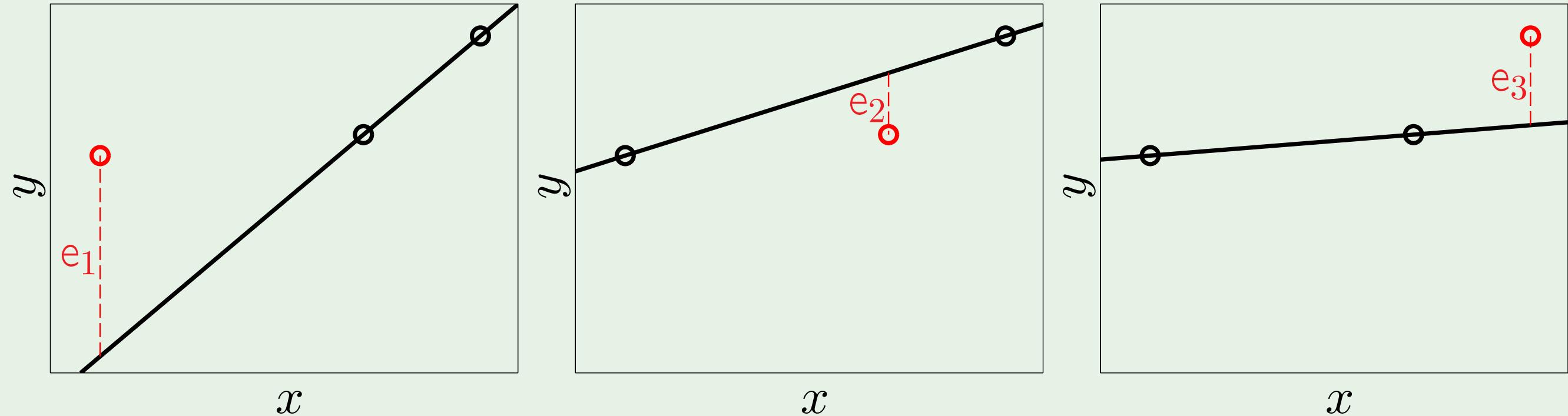
$$\mathcal{D}_n = (\mathbf{x}_1, y_1), \dots, (\mathbf{x}_{n-1}, y_{n-1}), (\cancel{\mathbf{x}_n, y_n}), (\mathbf{x}_{n+1}, y_{n+1}), \dots, (\mathbf{x}_N, y_N)$$

Final hypothesis learned from \mathcal{D}_n is g_n^-

$$\mathbf{e}_n = E_{\text{val}}(g_n^-) = \mathbf{e}(g_n^-(\mathbf{x}_n), y_n)$$

cross validation error: $E_{\text{cv}} = \frac{1}{N} \sum_{n=1}^N \mathbf{e}_n$

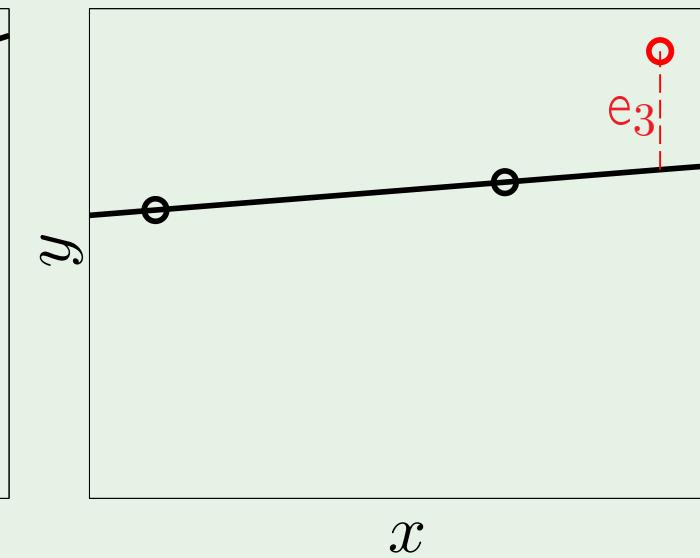
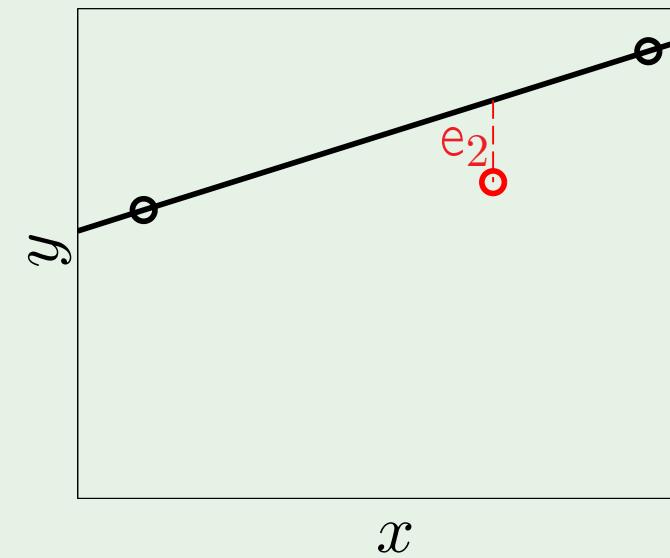
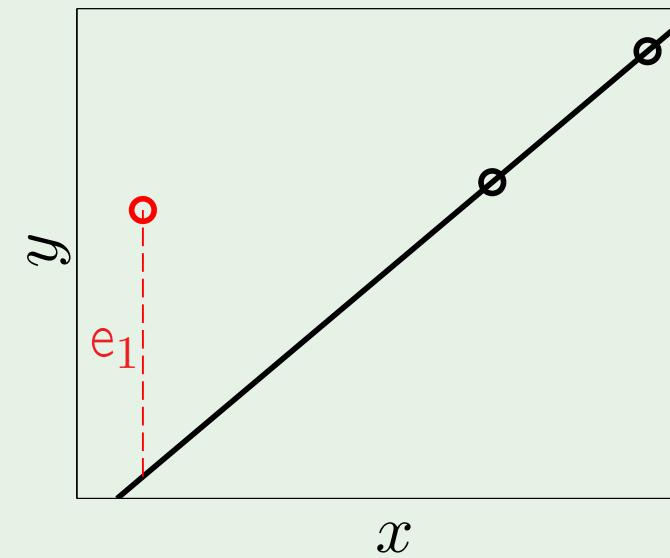
Illustration of cross validation



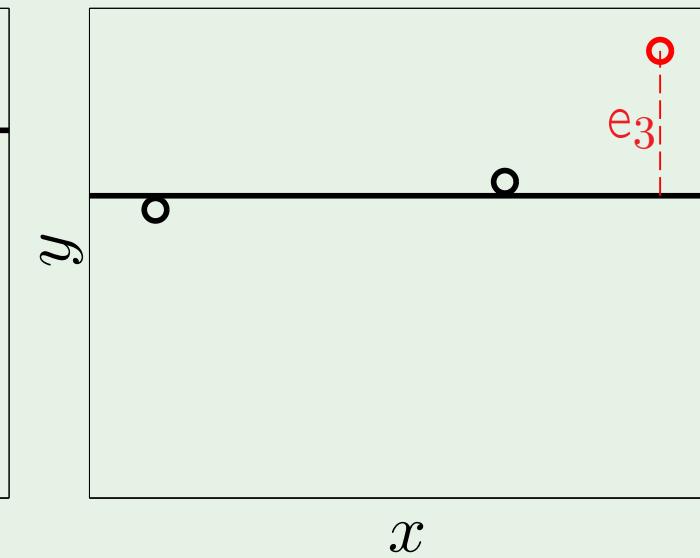
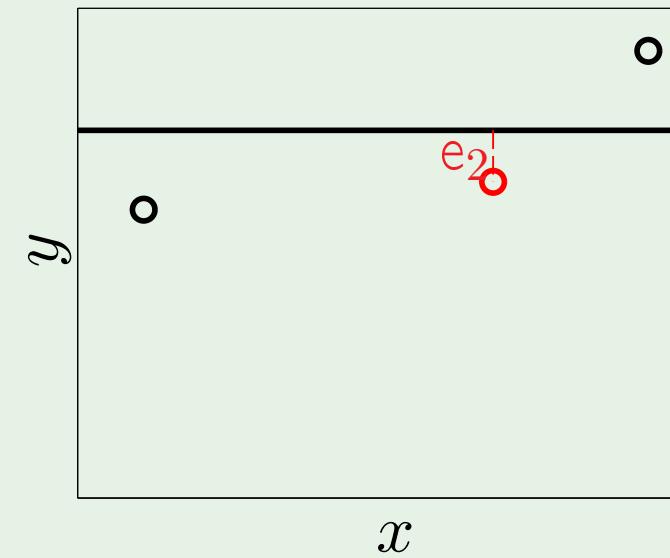
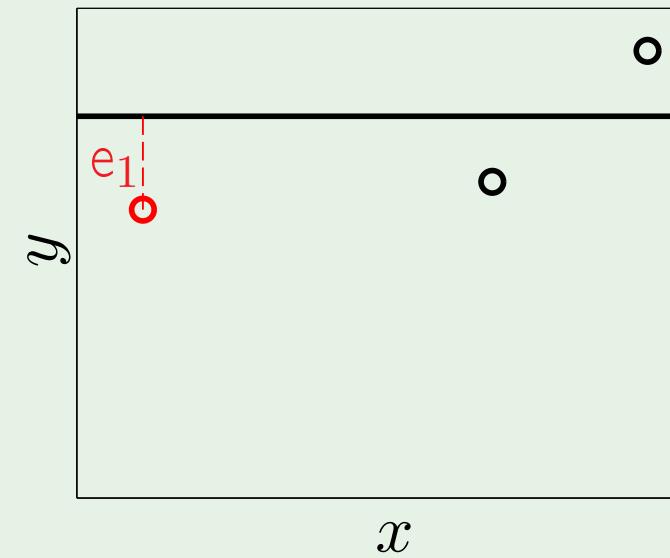
$$E_{\text{cv}} = \frac{1}{3} (e_1 + e_2 + e_3)$$

Model selection using CV

Linear:

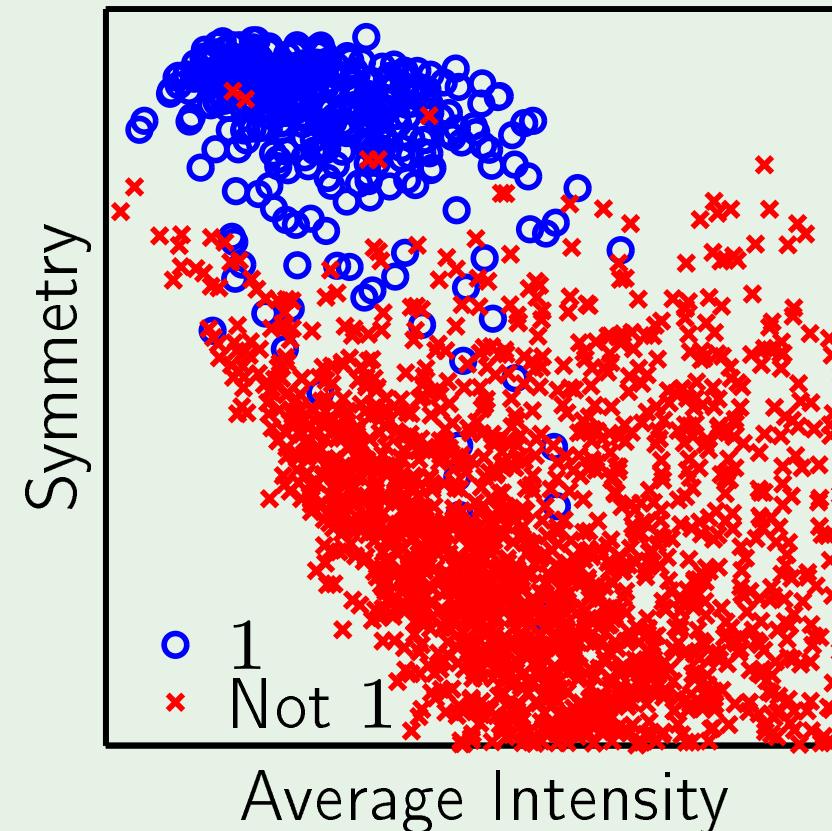


Constant:

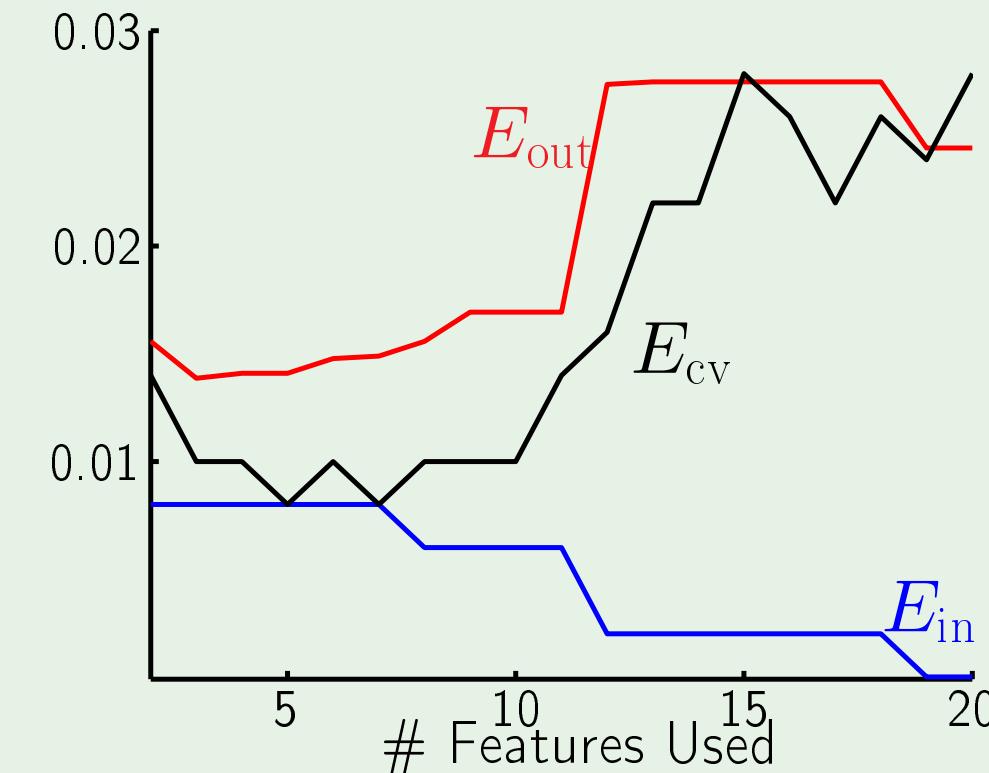


Cross validation in action

Digits classification task



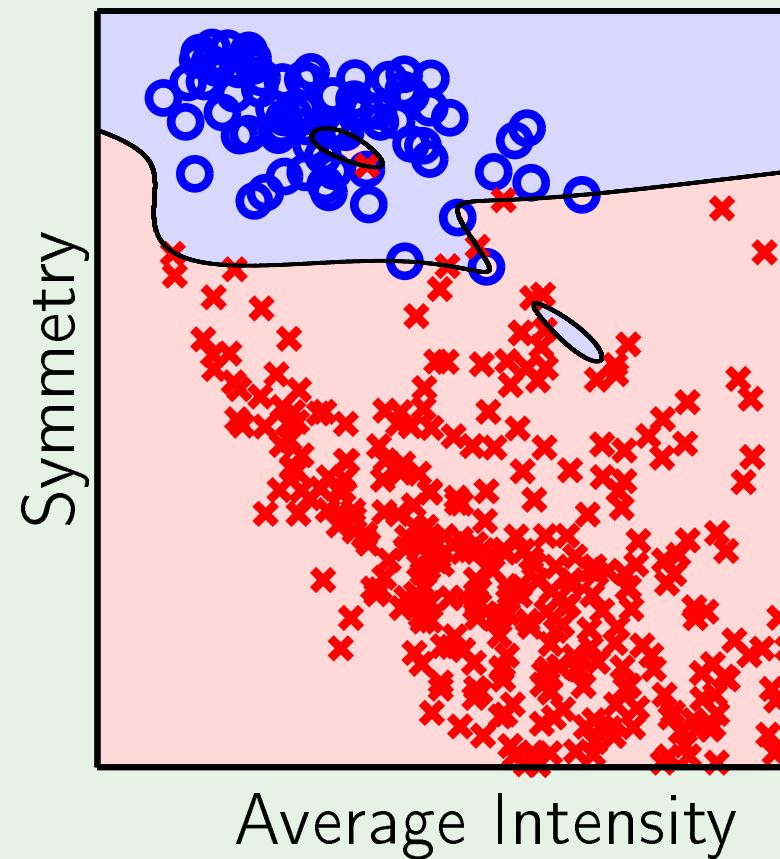
Different errors



$$(1, x_1, x_2) \rightarrow (1, x_1, x_2, x_1^2, x_1x_2, x_2^2, x_1^3, x_1^2x_2, \dots, x_1^5, x_1^4x_2, x_1^3x_2^2, x_1^2x_2^3, x_1x_2^4, x_2^5)$$

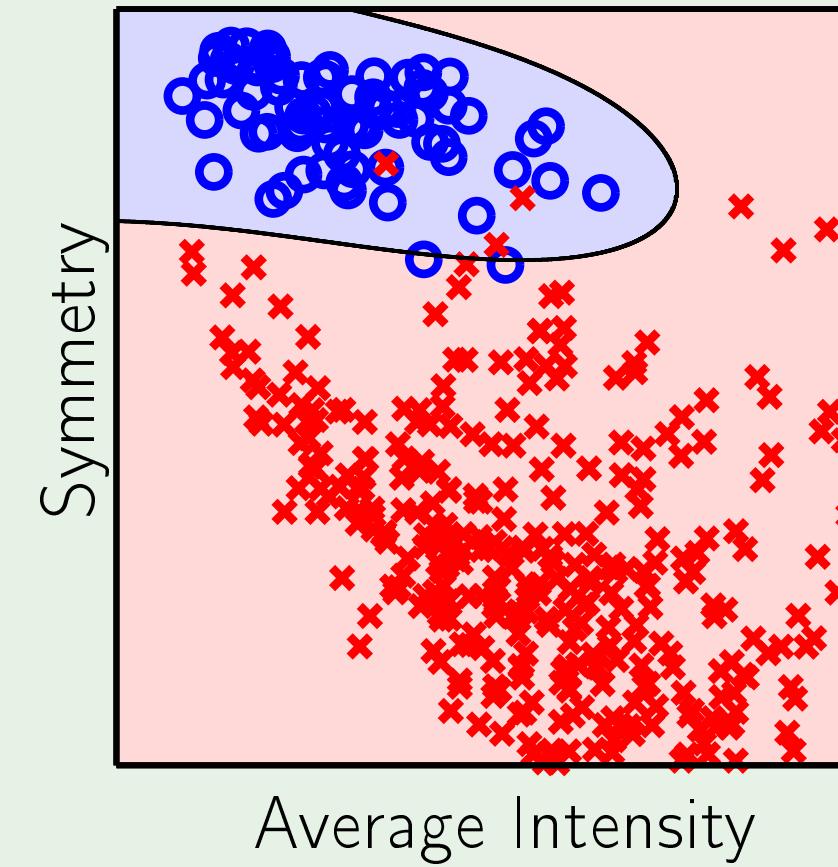
The result

without validation



$$E_{in} = 0\% \quad E_{out} = 2.5\%$$

with validation

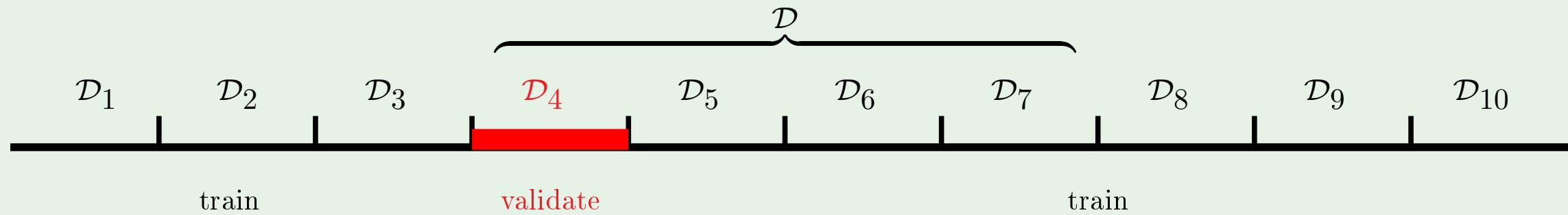


$$E_{in} = 0.8\% \quad E_{out} = 1.5\%$$

Leave more than one out

Leave one out: N training sessions on $N - 1$ points each

More points for validation?

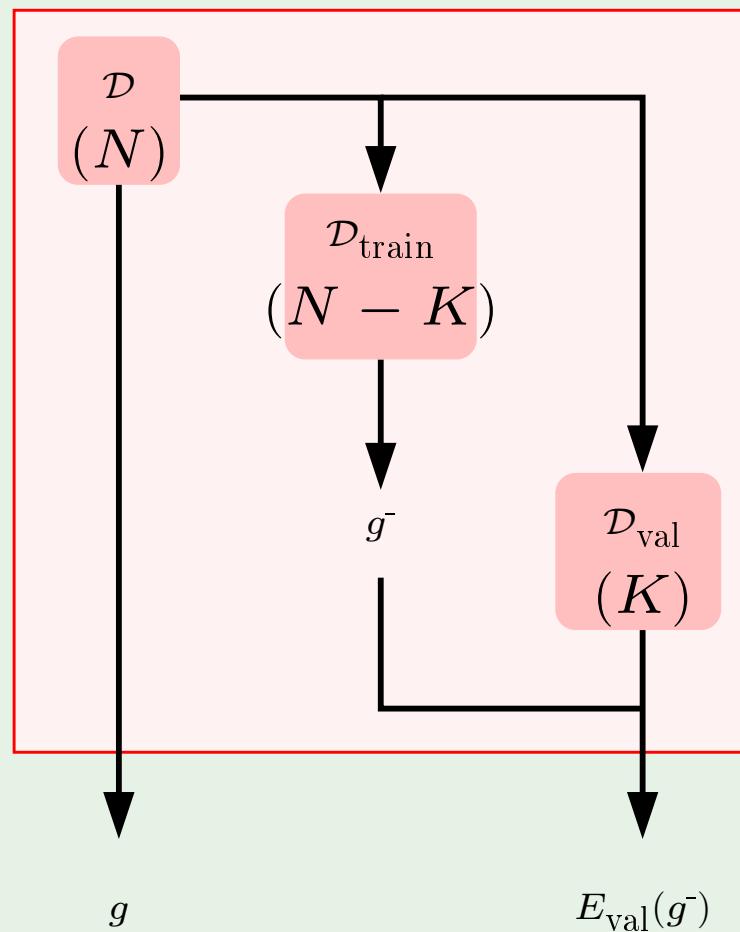


$\frac{N}{K}$ training sessions on $N - K$ points each

10-fold cross validation: $K = \frac{N}{10}$

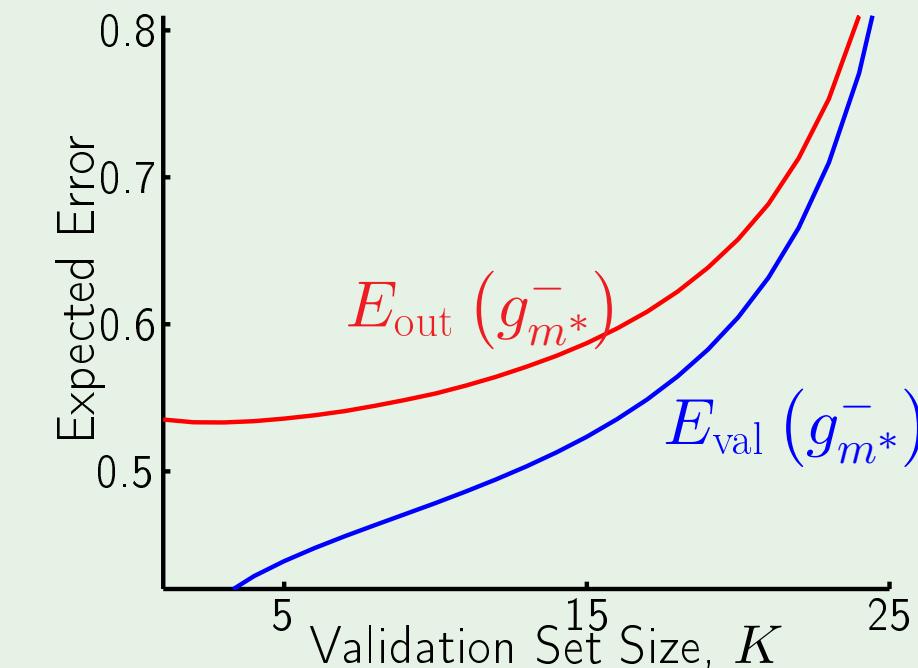
Review of Lecture 13

- Validation



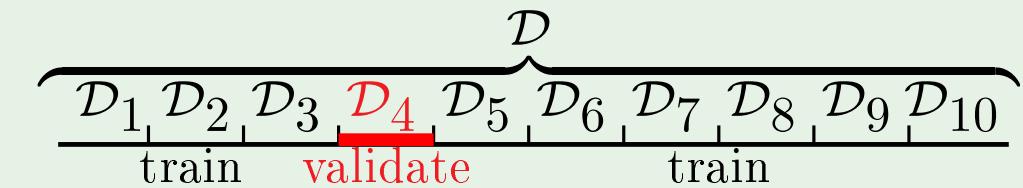
$E_{\text{val}}(g^-)$ estimates $E_{\text{out}}(g)$

- Data contamination



\mathcal{D}_{val} slightly contaminated

- Cross validation



10-fold cross validation

Learning From Data

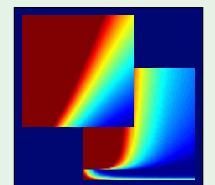
Yaser S. Abu-Mostafa
California Institute of Technology

Lecture 14: Support Vector Machines



Sponsored by Caltech's Provost Office, E&AS Division, and IST

• Thursday, May 17, 2012

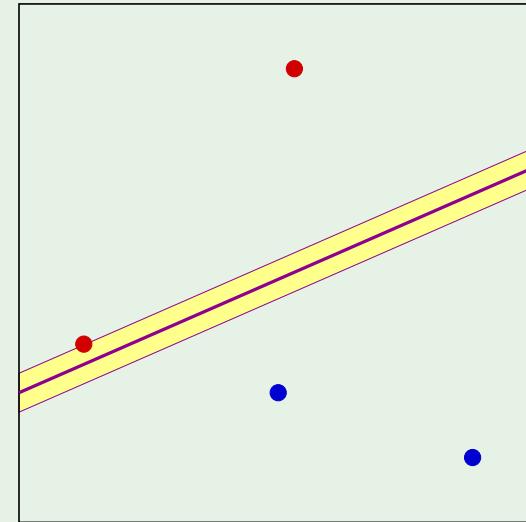


Outline

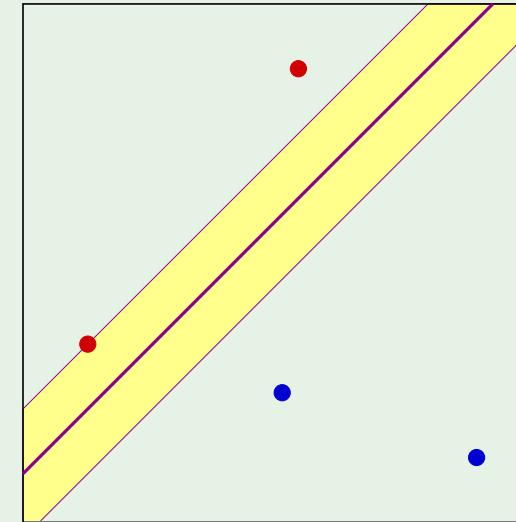
- Maximizing the margin
- The solution
- Nonlinear transforms

Better linear separation

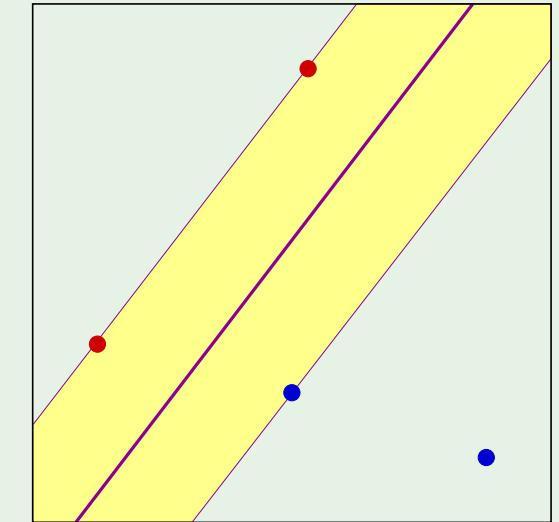
Linearly separable data



Different separating lines



Which is best?

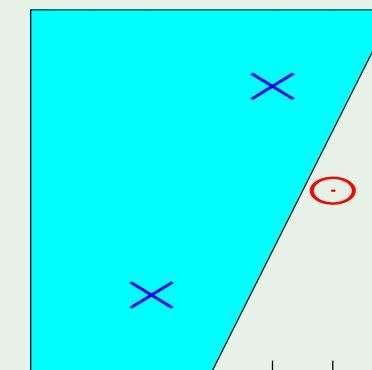
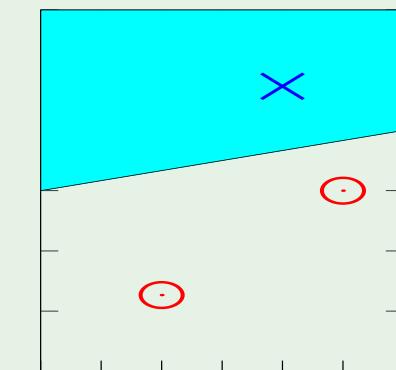
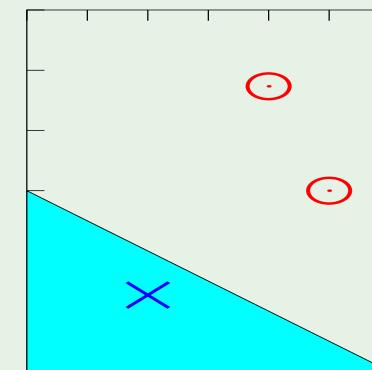
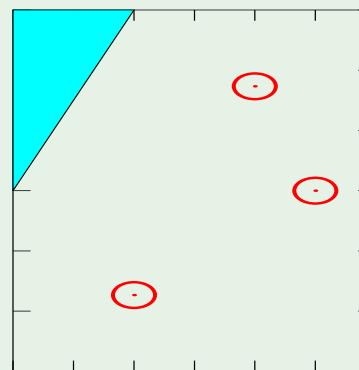
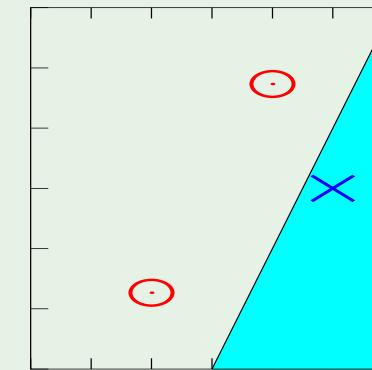
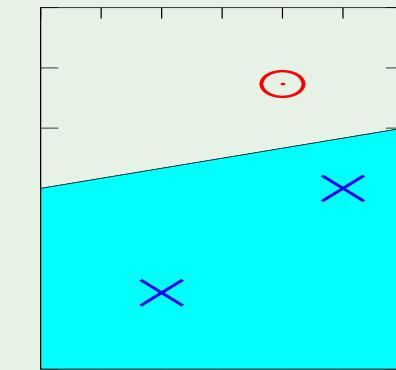
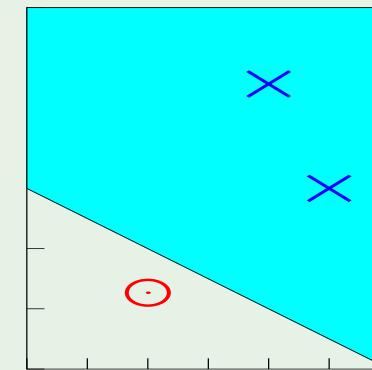
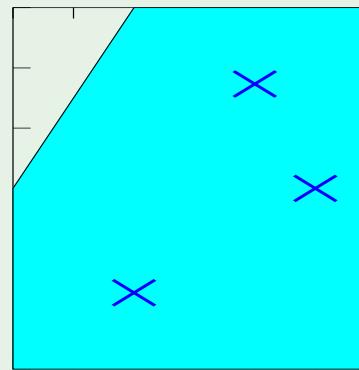


Two questions:

1. Why is bigger margin better?
2. Which \mathbf{w} maximizes the margin?

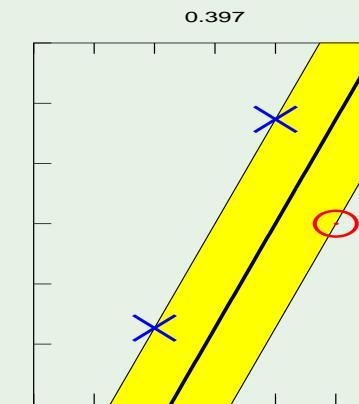
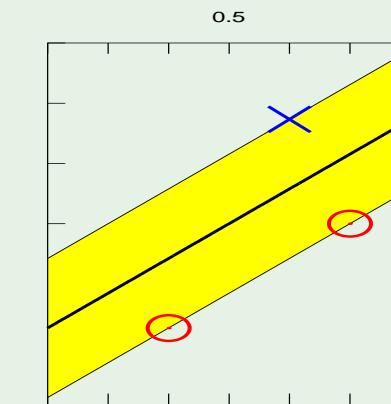
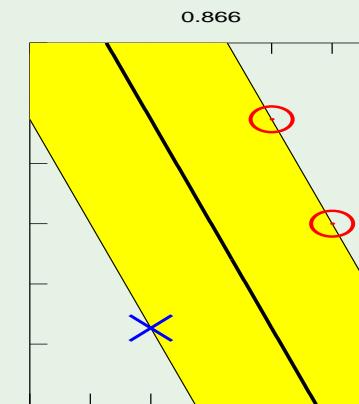
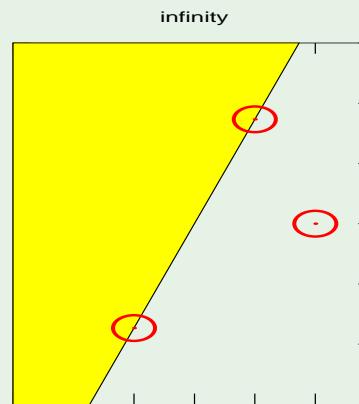
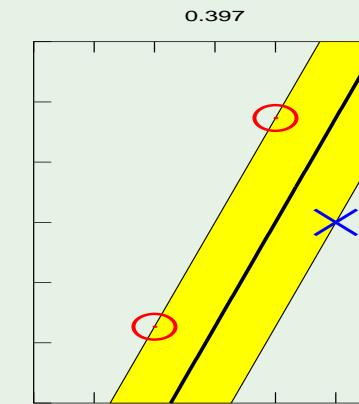
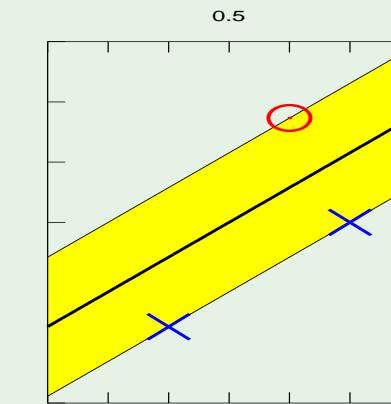
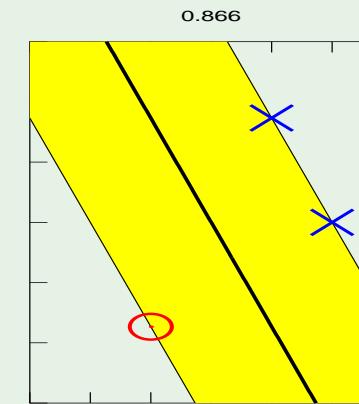
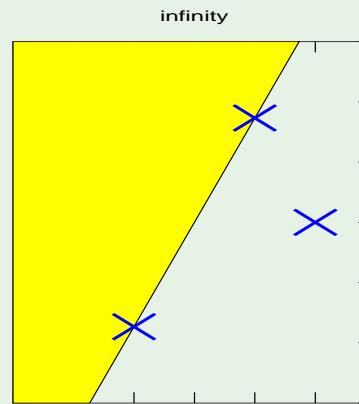
Remember the growth function?

All dichotomies with any line:



Dichotomies with fat margin

Fat margins imply fewer dichotomies



Finding \mathbf{w} with large margin

Let \mathbf{x}_n be the nearest data point to the plane $\mathbf{w}^\top \mathbf{x} = 0$. How far is it?

2 preliminary technicalities:

1. Normalize \mathbf{w} :

$$|\mathbf{w}^\top \mathbf{x}_n| = 1$$

2. Pull out w_0 :

$$\mathbf{w} = (w_1, \dots, w_d) \text{ apart from } b$$

The plane is now $\boxed{\mathbf{w}^\top \mathbf{x} + b = 0}$ (no x_0)

Computing the distance

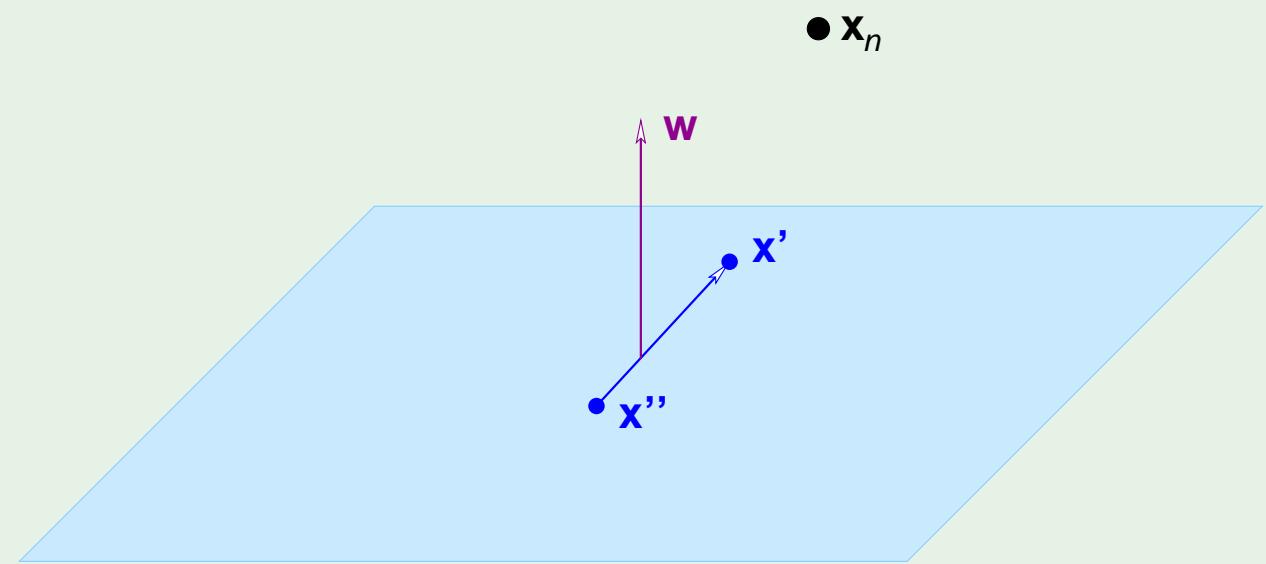
The distance between \mathbf{x}_n and the plane $\mathbf{w}^\top \mathbf{x} + b = 0$ where $|\mathbf{w}^\top \mathbf{x}_n + b| = 1$

The vector \mathbf{w} is \perp to the plane in the \mathcal{X} space:

Take \mathbf{x}' and \mathbf{x}'' on the plane

$$\mathbf{w}^\top \mathbf{x}' + b = 0 \quad \text{and} \quad \mathbf{w}^\top \mathbf{x}'' + b = 0$$

$$\implies \mathbf{w}^\top (\mathbf{x}' - \mathbf{x}'') = 0$$



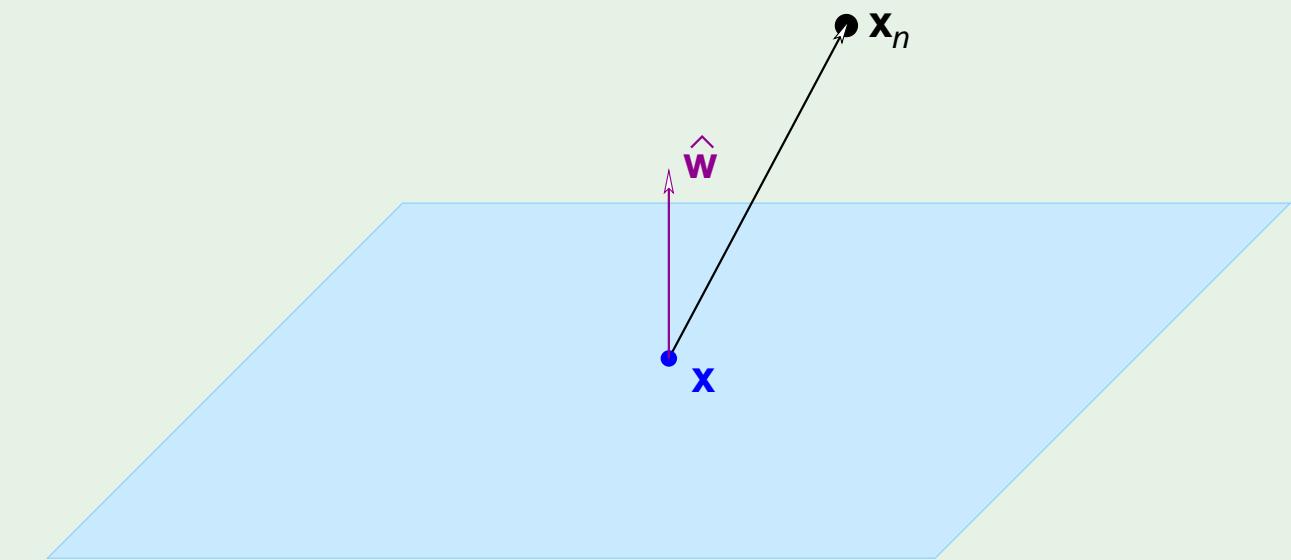
and the distance is ...

Distance between \mathbf{x}_n and the plane:

Take any point \mathbf{x} on the plane

Projection of $\mathbf{x}_n - \mathbf{x}$ on \mathbf{w}

$$\hat{\mathbf{w}} = \frac{\mathbf{w}}{\|\mathbf{w}\|} \implies \text{distance} = |\hat{\mathbf{w}}^\top(\mathbf{x}_n - \mathbf{x})|$$



$$\text{distance} = \frac{1}{\|\mathbf{w}\|} |\mathbf{w}^\top \mathbf{x}_n - \mathbf{w}^\top \mathbf{x}| = \frac{1}{\|\mathbf{w}\|} |\mathbf{w}^\top \mathbf{x}_n + b - \mathbf{w}^\top \mathbf{x} - b| = \frac{1}{\|\mathbf{w}\|}$$

The optimization problem

$$\text{Maximize } \frac{1}{\|\mathbf{w}\|}$$

$$\text{subject to } \min_{n=1,2,\dots,N} |\mathbf{w}^\top \mathbf{x}_n + b| = 1$$

Notice: $|\mathbf{w}^\top \mathbf{x}_n + b| = y_n (\mathbf{w}^\top \mathbf{x}_n + b)$

$$\text{Minimize } \frac{1}{2} \mathbf{w}^\top \mathbf{w}$$

$$\text{subject to } y_n (\mathbf{w}^\top \mathbf{x}_n + b) \geq 1 \quad \text{for } n = 1, 2, \dots, N$$

Outline

- Maximizing the margin
- The solution
- Nonlinear transforms

Constrained optimization

$$\text{Minimize} \quad \frac{1}{2} \mathbf{w}^\top \mathbf{w}$$

$$\text{subject to} \quad y_n (\mathbf{w}^\top \mathbf{x}_n + b) \geq 1 \quad \text{for } n = 1, 2, \dots, N$$

$$\mathbf{w} \in \mathbb{R}^d, b \in \mathbb{R}$$

Lagrange? inequality constraints \implies KKT

We saw this before

Remember regularization?

$$\text{Minimize } E_{\text{in}}(\mathbf{w}) = \frac{1}{N} (\mathbf{Z}\mathbf{w} - \mathbf{y})^\top (\mathbf{Z}\mathbf{w} - \mathbf{y})$$

$$\text{subject to: } \mathbf{w}^\top \mathbf{w} \leq C$$

∇E_{in} normal to constraint

optimize

constrain

Regularization:

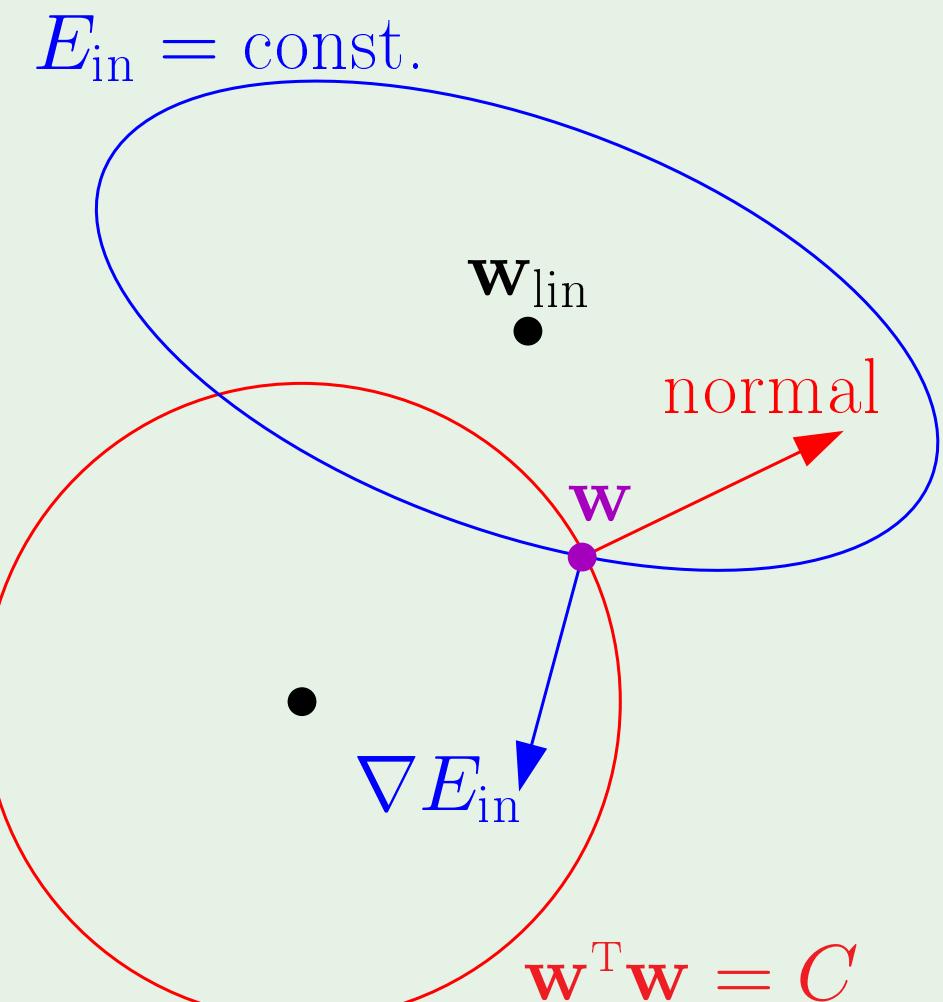
$$E_{\text{in}}$$

$$\mathbf{w}^\top \mathbf{w}$$

SVM:

$$\mathbf{w}^\top \mathbf{w}$$

$$E_{\text{in}}$$



Lagrange formulation

$$\text{Minimize } \mathcal{L}(\mathbf{w}, b, \alpha) = \frac{1}{2} \mathbf{w}^\top \mathbf{w} - \sum_{n=1}^N \alpha_n (y_n (\mathbf{w}^\top \mathbf{x}_n + b) - 1)$$

w.r.t. \mathbf{w} and b and maximize w.r.t. each $\alpha_n \geq 0$

$$\nabla_{\mathbf{w}} \mathcal{L} = \mathbf{w} - \sum_{n=1}^N \alpha_n y_n \mathbf{x}_n = 0$$

$$\frac{\partial \mathcal{L}}{\partial b} = - \sum_{n=1}^N \alpha_n y_n = 0$$

Substituting ...

$$\mathbf{w} = \sum_{n=1}^N \alpha_n y_n \mathbf{x}_n \quad \text{and} \quad \sum_{n=1}^N \alpha_n y_n = 0$$

in the Lagrangian

$$\mathcal{L}(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2} \mathbf{w}^\top \mathbf{w} - \sum_{n=1}^N \alpha_n (y_n (\mathbf{w}^\top \mathbf{x}_n + b) - 1)$$

we get

$$\mathcal{L}(\boldsymbol{\alpha}) = \sum_{n=1}^N \alpha_n - \frac{1}{2} \sum_{n=1}^N \sum_{m=1}^N y_n y_m \alpha_n \alpha_m \mathbf{x}_n^\top \mathbf{x}_m$$

Maximize w.r.t. to $\boldsymbol{\alpha}$ subject to $\alpha_n \geq 0$ for $n = 1, \dots, N$ and $\sum_{n=1}^N \alpha_n y_n = 0$

The solution - quadratic programming

$$\min_{\alpha} \frac{1}{2} \alpha^\top \underbrace{\begin{bmatrix} y_1 y_1 \mathbf{x}_1^\top \mathbf{x}_1 & y_1 y_2 \mathbf{x}_1^\top \mathbf{x}_2 & \dots & y_1 y_N \mathbf{x}_1^\top \mathbf{x}_N \\ y_2 y_1 \mathbf{x}_2^\top \mathbf{x}_1 & y_2 y_2 \mathbf{x}_2^\top \mathbf{x}_2 & \dots & y_2 y_N \mathbf{x}_2^\top \mathbf{x}_N \\ \dots & \dots & \dots & \dots \\ y_N y_1 \mathbf{x}_N^\top \mathbf{x}_1 & y_N y_2 \mathbf{x}_N^\top \mathbf{x}_2 & \dots & y_N y_N \mathbf{x}_N^\top \mathbf{x}_N \end{bmatrix}}_{\text{quadratic coefficients}} \alpha + \underbrace{(-1^\top) \alpha}_{\text{linear}}$$

subject to

$$\underbrace{\mathbf{y}^\top \alpha = 0}_{\text{linear constraint}}$$

$$\underbrace{0}_{\text{lower bounds}} \leq \alpha \leq \underbrace{\infty}_{\text{upper bounds}}$$

QP hands us α

Solution: $\alpha = \alpha_1, \dots, \alpha_N$

$$\Rightarrow \mathbf{w} = \sum_{n=1}^N \alpha_n y_n \mathbf{x}_n$$

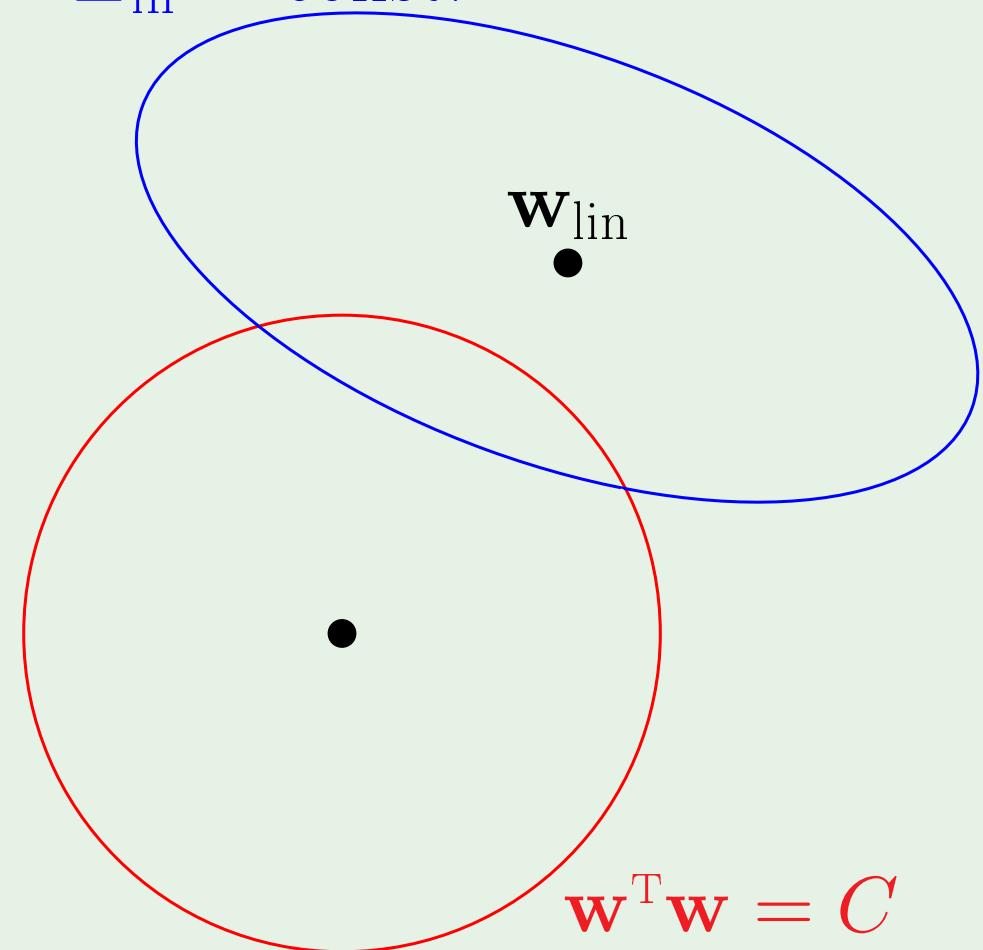
KKT condition: For $n = 1, \dots, N$

$$\alpha_n (y_n (\mathbf{w}^\top \mathbf{x}_n + b) - 1) = 0$$

We saw this before!

$\alpha_n > 0 \implies \mathbf{x}_n$ is a **support vector**

$$E_{\text{in}} = \text{const.}$$



Support vectors

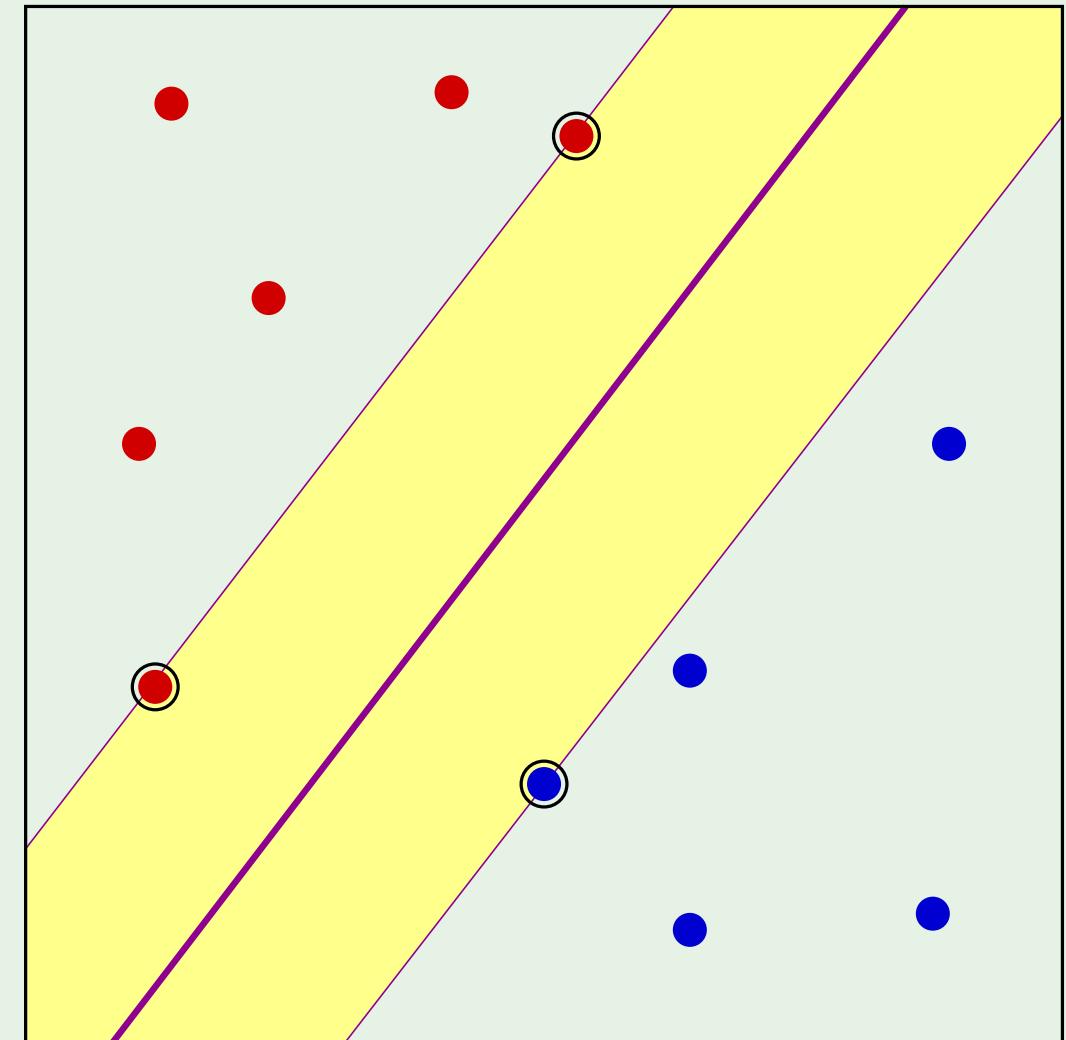
Closest \mathbf{x}_n 's to the plane: achieve the margin

$$\implies y_n (\mathbf{w}^\top \mathbf{x}_n + b) = 1$$

$$\mathbf{w} = \sum_{\mathbf{x}_n \text{ is SV}} \alpha_n y_n \mathbf{x}_n$$

Solve for b using any SV:

$$y_n (\mathbf{w}^\top \mathbf{x}_n + b) = 1$$

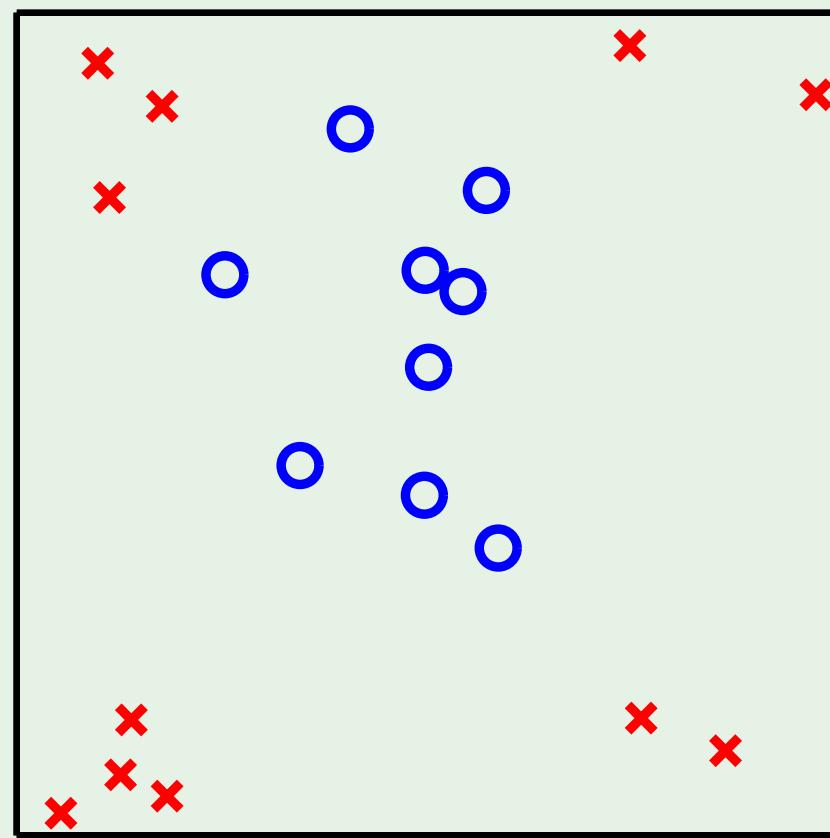


Outline

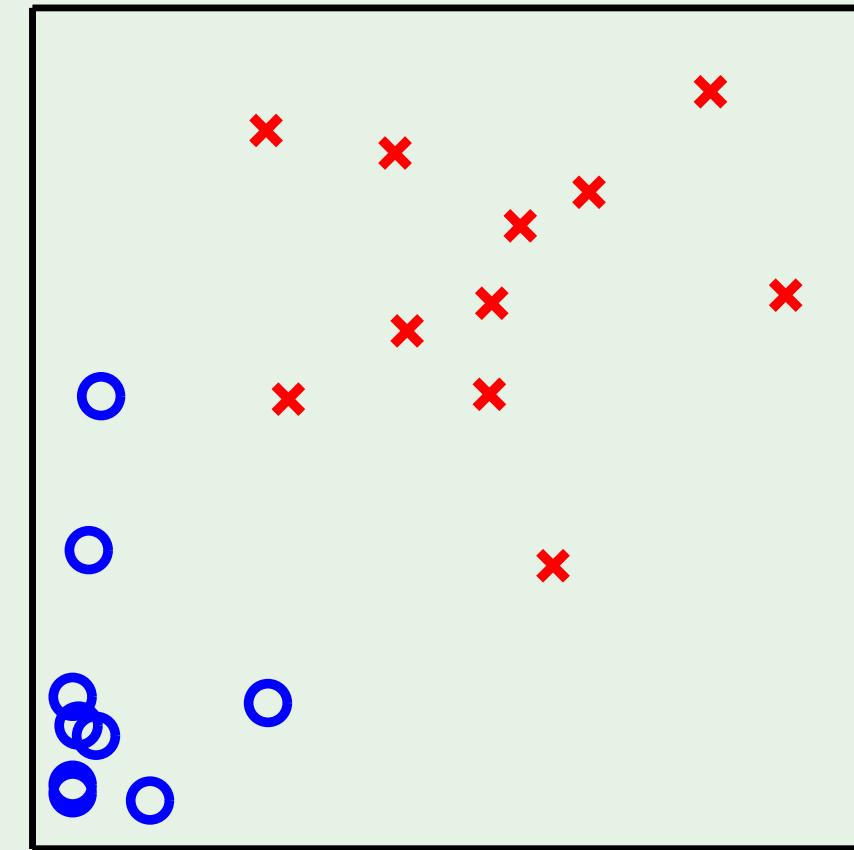
- Maximizing the margin
- The solution
- Nonlinear transforms

z instead of x

$$\mathcal{L}(\alpha) = \sum_{n=1}^N \alpha_n - \frac{1}{2} \sum_{n=1}^N \sum_{m=1}^N y_n y_m \alpha_n \alpha_m \mathbf{z}_n^\top \mathbf{z}_m$$



$\mathcal{X} \rightarrow \mathcal{Z}$



“Support vectors” in \mathcal{X} space

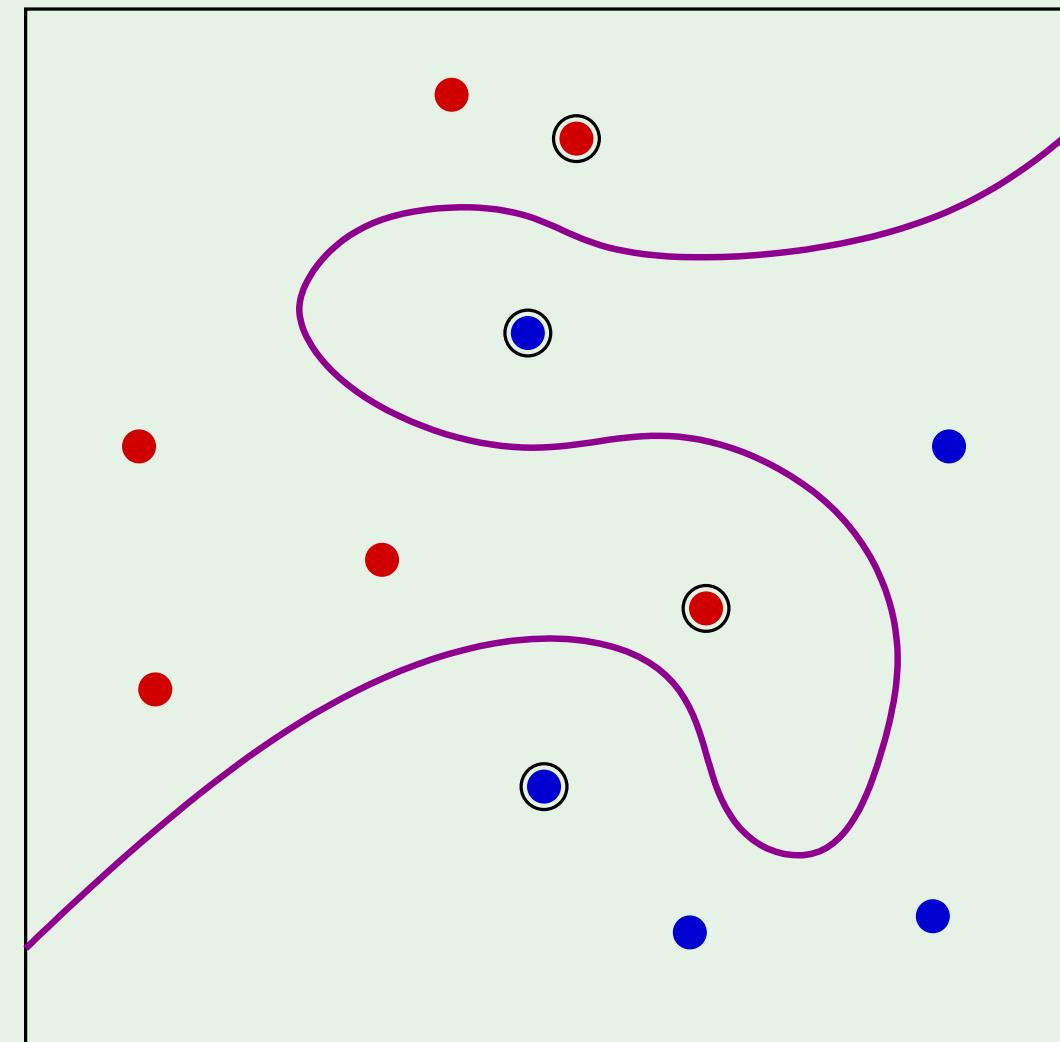
Support vectors live in \mathcal{Z} space

In \mathcal{X} space, “pre-images” of support vectors

The margin is maintained in \mathcal{Z} space

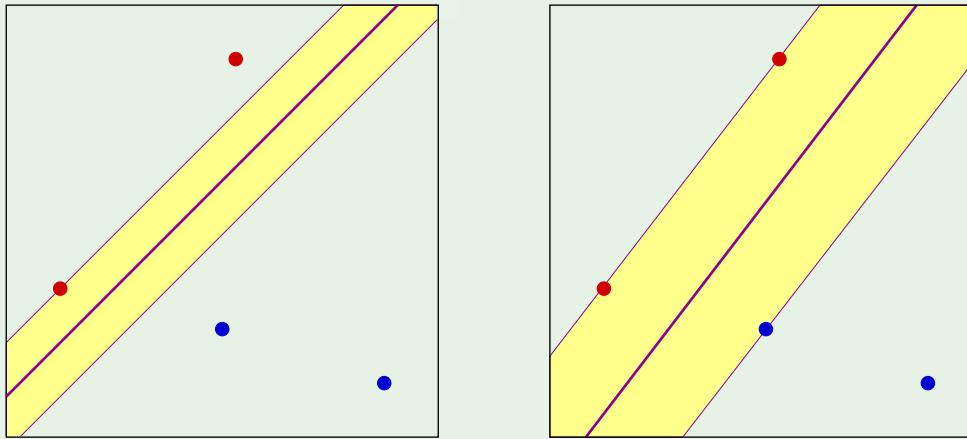
Generalization result

$$\mathbb{E}[E_{\text{out}}] \leq \frac{\mathbb{E}[\# \text{ of SV's}]}{N - 1}$$



Review of Lecture 14

- The margin

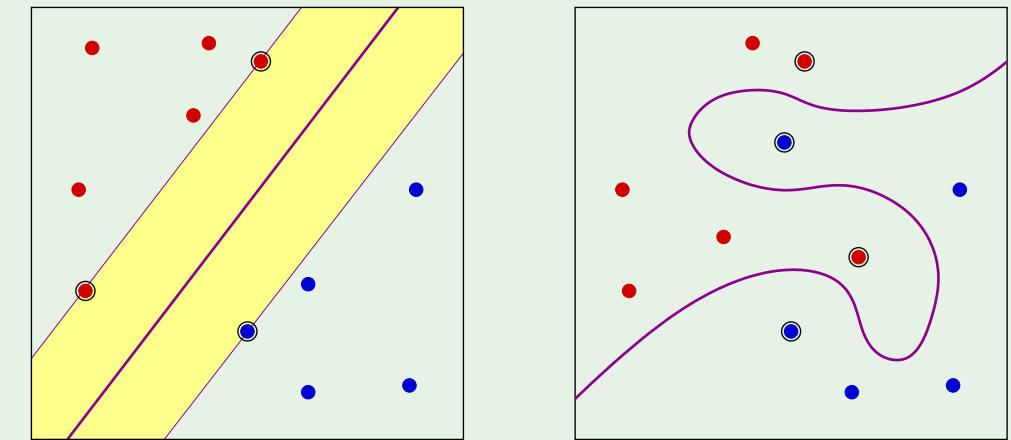


Maximizing the margin \implies dual problem:

$$\mathcal{L}(\alpha) = \sum_{n=1}^N \alpha_n - \frac{1}{2} \sum_{n=1}^N \sum_{m=1}^N y_n y_m \alpha_n \alpha_m \mathbf{x}_n^\top \mathbf{x}_m$$

quadratic programming

- Support vectors



\mathbf{x}_n (or \mathbf{z}_n) with Lagrange $\alpha_n > 0$

$$\mathbb{E}[E_{\text{out}}] \leq \frac{\mathbb{E}[\# \text{ of SV's}]}{N - 1}$$

(in-sample check of out-of-sample error)

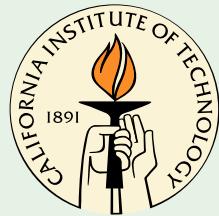
- Nonlinear transform

Complex h , but simple \mathcal{H} ☺

Learning From Data

Yaser S. Abu-Mostafa
California Institute of Technology

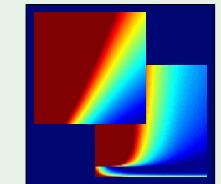
Lecture 15: Kernel Methods



Sponsored by Caltech's Provost Office, E&AS Division, and IST



Tuesday, May 22, 2012



Outline

- The kernel trick
- Soft-margin SVM

What do we need from the \mathcal{Z} space?

$$\mathcal{L}(\boldsymbol{\alpha}) = \sum_{n=1}^N \alpha_n - \frac{1}{2} \sum_{n=1}^N \sum_{m=1}^N y_n y_m \alpha_n \alpha_m \mathbf{z}_n^\top \mathbf{z}_m$$

Constraints: $\alpha_n \geq 0$ for $n = 1, \dots, N$ and $\sum_{n=1}^N \alpha_n y_n = 0$

$$g(\mathbf{x}) = \text{sign} (\mathbf{w}^\top \mathbf{z} + b) \quad \text{need } \mathbf{z}_n^\top \mathbf{z}$$

where $\mathbf{w} = \sum_{\mathbf{z}_n \text{ is SV}} \alpha_n y_n \mathbf{z}_n$

and b : $y_m (\mathbf{w}^\top \mathbf{z}_m + b) = 1$ need $\mathbf{z}_n^\top \mathbf{z}_m$

Generalized inner product

Given two points \mathbf{x} and $\mathbf{x}' \in \mathcal{X}$, we need $\mathbf{z}^\top \mathbf{z}'$

Let $\mathbf{z}^\top \mathbf{z}' = K(\mathbf{x}, \mathbf{x}')$ (the kernel) “inner product” of \mathbf{x} and \mathbf{x}'

Example: $\mathbf{x} = (x_1, x_2) \longrightarrow$ 2nd-order Φ

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

$$K(\mathbf{x}, \mathbf{x}') = \mathbf{z}^\top \mathbf{z}' = 1 + x_1 x'_1 + x_2 x'_2 +$$

$$x_1^2 x'^2_1 + x_2^2 x'^2_2 + x_1 x'_1 x_2 x'_2$$

The trick

Can we compute $K(\mathbf{x}, \mathbf{x}')$ **without** transforming \mathbf{x} and \mathbf{x}' ?

Example: Consider $K(\mathbf{x}, \mathbf{x}') = (1 + \mathbf{x}^\top \mathbf{x}')^2 = (1 + x_1 x'_1 + x_2 x'_2)^2$

$$= 1 + x_1^2 x'^2_1 + x_2^2 x'^2_2 + 2x_1 x'_1 + 2x_2 x'_2 + 2x_1 x'_1 x_2 x'_2$$

This is an inner product!

$$(1, x_1^2, x_2^2, \sqrt{2}x_1, \sqrt{2}x_2, \sqrt{2}x_1 x_2)$$

$$(1, x'^2_1, x'^2_2, \sqrt{2}x'_1, \sqrt{2}x'_2, \sqrt{2}x'_1 x'_2)$$

The polynomial kernel

$\mathcal{X} = \mathbb{R}^d$ and $\Phi : \mathcal{X} \rightarrow \mathcal{Z}$ is polynomial of order Q

The “equivalent” kernel $K(\mathbf{x}, \mathbf{x}') = (1 + \mathbf{x}^\top \mathbf{x}')^Q$

$$= (1 + x_1 x'_1 + x_2 x'_2 + \cdots + x_d x'_d)^Q$$

Compare for $d = 10$ and $Q = 100$

Can adjust scale: $K(\mathbf{x}, \mathbf{x}') = (a \mathbf{x}^\top \mathbf{x}' + b)^Q$

We only need \mathcal{Z} to exist!

If $K(\mathbf{x}, \mathbf{x}')$ is an inner product in some space \mathcal{Z} , we are good.

Example:
$$K(\mathbf{x}, \mathbf{x}') = \exp(-\gamma \|\mathbf{x} - \mathbf{x}'\|^2)$$

Infinite-dimensional \mathcal{Z} : take simple case

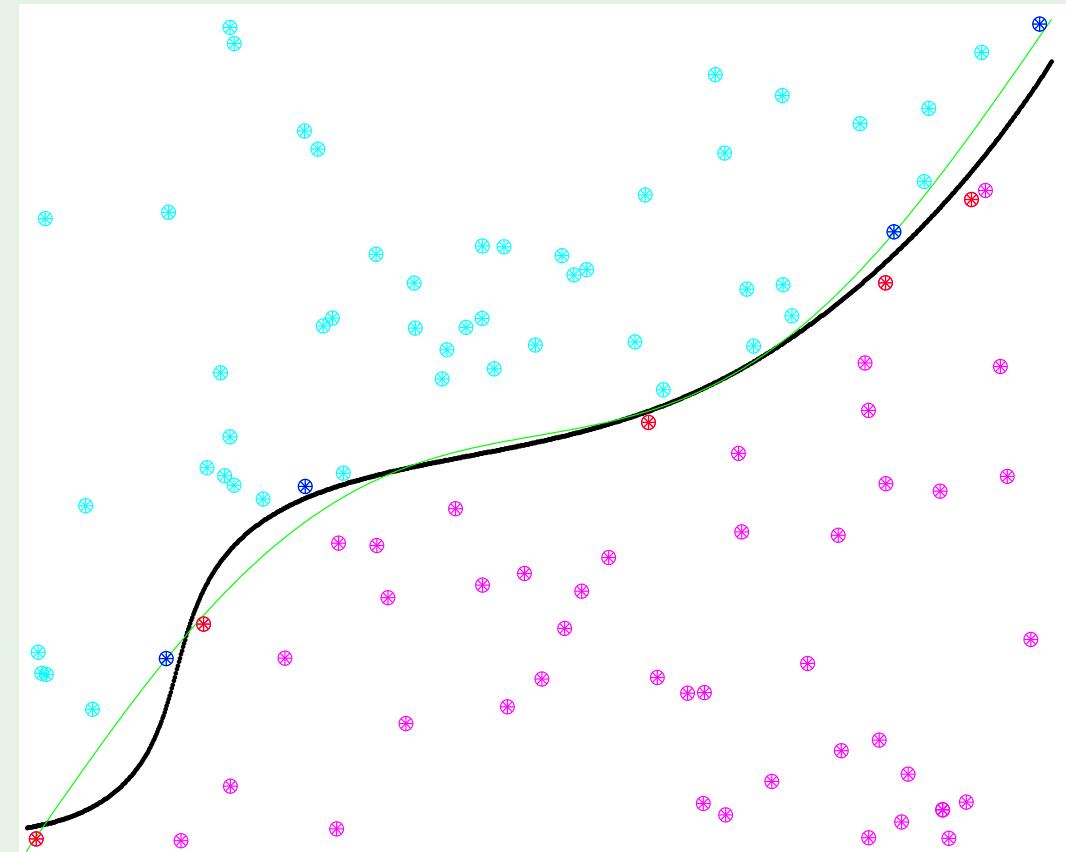
$$\begin{aligned} K(x, x') &= \exp(-(x - x')^2) \\ &= \exp(-x^2) \exp(-x'^2) \underbrace{\sum_{k=0}^{\infty} \frac{2^k (x)^k (x')^k}{k!}}_{\exp(2xx')} \end{aligned}$$

This kernel in action

Slightly non-separable case:

Transforming \mathcal{X} into ∞ -dimensional \mathcal{Z}

Overkill? Count the support vectors



Kernel formulation of SVM

Remember quadratic programming? The only difference now is:

$$\begin{bmatrix} y_1y_1 K(\mathbf{x}_1, \mathbf{x}_1) & y_1y_2 K(\mathbf{x}_1, \mathbf{x}_2) & \dots & y_1y_N K(\mathbf{x}_1, \mathbf{x}_N) \\ y_2y_1 K(\mathbf{x}_2, \mathbf{x}_1) & y_2y_2 K(\mathbf{x}_2, \mathbf{x}_2) & \dots & y_2y_N K(\mathbf{x}_2, \mathbf{x}_N) \\ \dots & \dots & \dots & \dots \\ y_Ny_1 K(\mathbf{x}_N, \mathbf{x}_1) & y_Ny_2 K(\mathbf{x}_N, \mathbf{x}_2) & \dots & y_Ny_N K(\mathbf{x}_N, \mathbf{x}_N) \end{bmatrix}$$

$\underbrace{\qquad\qquad\qquad}_{\text{quadratic coefficients}}$

Everything else is the same.

The final hypothesis

Express $g(\mathbf{x}) = \text{sign} (\mathbf{w}^\top \mathbf{z} + b)$ in terms of $K(-, -)$

$$\mathbf{w} = \sum_{\mathbf{z}_n \text{ is SV}} \alpha_n y_n \mathbf{z}_n \implies g(\mathbf{x}) = \text{sign} \left(\sum_{\alpha_n > 0} \alpha_n y_n K(\mathbf{x}_n, \mathbf{x}) + b \right)$$

$$\text{where } b = y_m - \sum_{\alpha_n > 0} \alpha_n y_n K(\mathbf{x}_n, \mathbf{x}_m)$$

for any support vector ($\alpha_m > 0$)

How do we know that \mathcal{Z} exists ...

... for a given $K(\mathbf{x}, \mathbf{x}')$? valid kernel

Three approaches:

1. By construction
2. Math properties (*Mercer's condition*)
3. Who cares? ☺

Design your own kernel

$K(\mathbf{x}, \mathbf{x}')$ is a valid kernel iff

1. It is symmetric and
2. The matrix:

$$\begin{bmatrix} K(\mathbf{x}_1, \mathbf{x}_1) & K(\mathbf{x}_1, \mathbf{x}_2) & \dots & K(\mathbf{x}_1, \mathbf{x}_N) \\ K(\mathbf{x}_2, \mathbf{x}_1) & K(\mathbf{x}_2, \mathbf{x}_2) & \dots & K(\mathbf{x}_2, \mathbf{x}_N) \\ \dots & \dots & \dots & \dots \\ K(\mathbf{x}_N, \mathbf{x}_1) & K(\mathbf{x}_N, \mathbf{x}_2) & \dots & K(\mathbf{x}_N, \mathbf{x}_N) \end{bmatrix}$$

is **positive semi-definite**

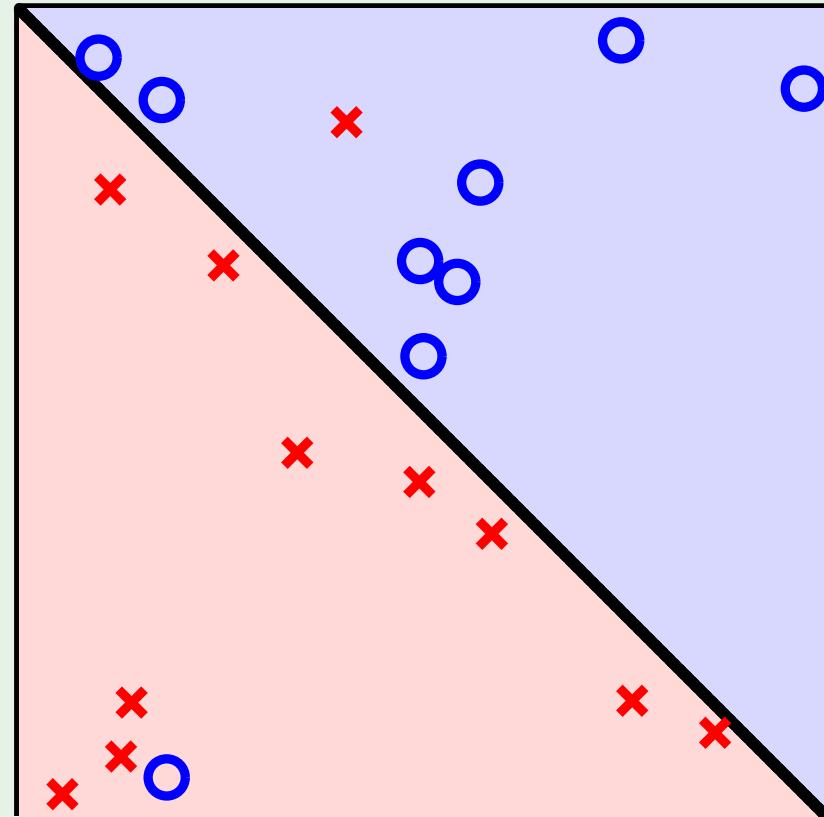
for any $\mathbf{x}_1, \dots, \mathbf{x}_N$ (Mercer's condition)

Outline

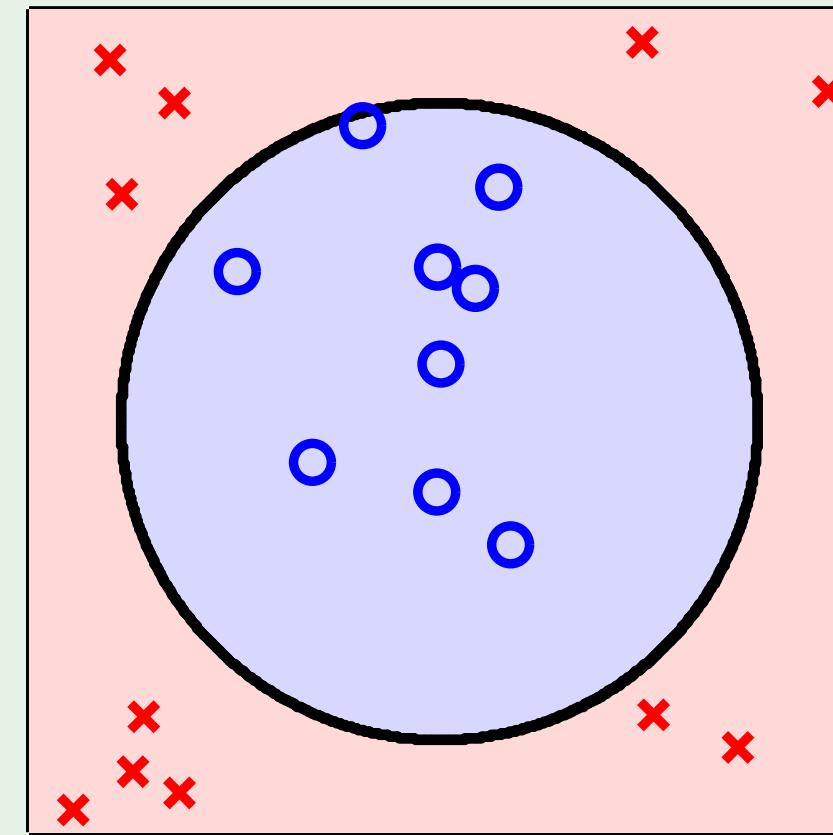
- The kernel trick
- Soft-margin SVM

Two types of non-separable

slightly:



seriously:

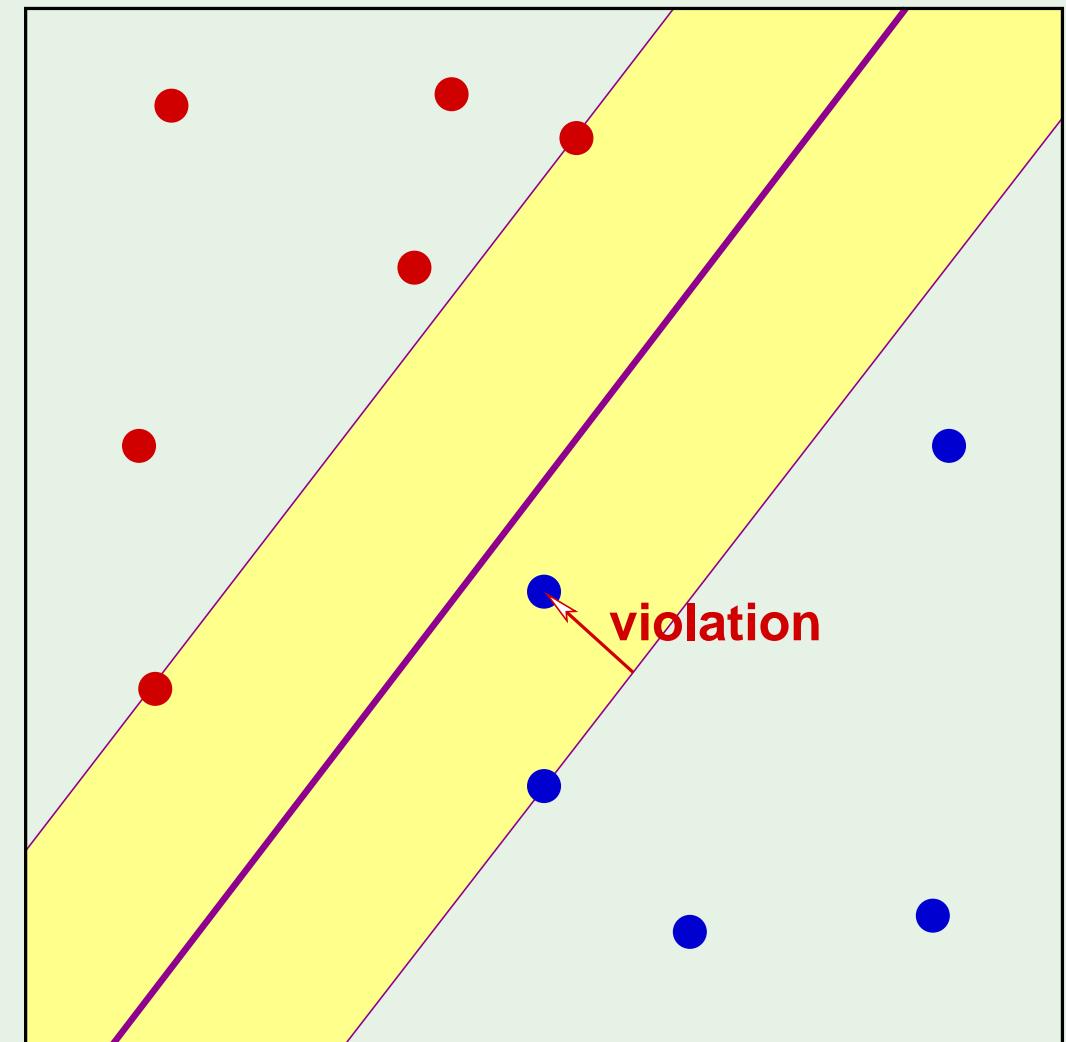


Error measure

Margin violation: $y_n (\mathbf{w}^\top \mathbf{x}_n + b) \geq 1$ fails

Quantify: $y_n (\mathbf{w}^\top \mathbf{x}_n + b) \geq 1 - \xi_n \quad \xi_n \geq 0$

$$\text{Total violation} = \sum_{n=1}^N \xi_n$$



The new optimization

Minimize $\frac{1}{2} \mathbf{w}^\top \mathbf{w} + C \sum_{n=1}^N \xi_n$

subject to $y_n (\mathbf{w}^\top \mathbf{x}_n + b) \geq 1 - \xi_n \quad \text{for } n = 1, \dots, N$

and $\xi_n \geq 0 \quad \text{for } n = 1, \dots, N$

$$\mathbf{w} \in \mathbb{R}^d , \quad b \in \mathbb{R} , \quad \boldsymbol{\xi} \in \mathbb{R}^N$$

Lagrange formulation

$$\mathcal{L}(\mathbf{w}, b, \xi, \alpha, \beta) = \frac{1}{2} \mathbf{w}^\top \mathbf{w} + C \sum_{n=1}^N \xi_n - \sum_{n=1}^N \alpha_n (y_n (\mathbf{w}^\top \mathbf{x}_n + b) - 1 + \xi_n) - \sum_{n=1}^N \beta_n \xi_n$$

Minimize w.r.t. \mathbf{w} , b , and ξ and maximize w.r.t. each $\alpha_n \geq 0$ and $\beta_n \geq 0$

$$\nabla_{\mathbf{w}} \mathcal{L} = \mathbf{w} - \sum_{n=1}^N \alpha_n y_n \mathbf{x}_n = 0$$

$$\frac{\partial \mathcal{L}}{\partial b} = - \sum_{n=1}^N \alpha_n y_n = 0$$

$$\frac{\partial \mathcal{L}}{\partial \xi_n} = C - \alpha_n - \beta_n = 0$$

and the solution is ...

Maximize $\mathcal{L}(\boldsymbol{\alpha}) = \sum_{n=1}^N \alpha_n - \frac{1}{2} \sum_{n=1}^N \sum_{m=1}^N y_n y_m \alpha_n \alpha_m \mathbf{x}_n^\top \mathbf{x}_m$ w.r.t. to $\boldsymbol{\alpha}$

subject to $0 \leq \alpha_n \leq C$ for $n = 1, \dots, N$ and $\sum_{n=1}^N \alpha_n y_n = 0$

$$\implies \mathbf{w} = \sum_{n=1}^N \alpha_n y_n \mathbf{x}_n$$

$$\text{minimizes } \frac{1}{2} \mathbf{w}^\top \mathbf{w} + C \sum_{n=1}^N \xi_n$$

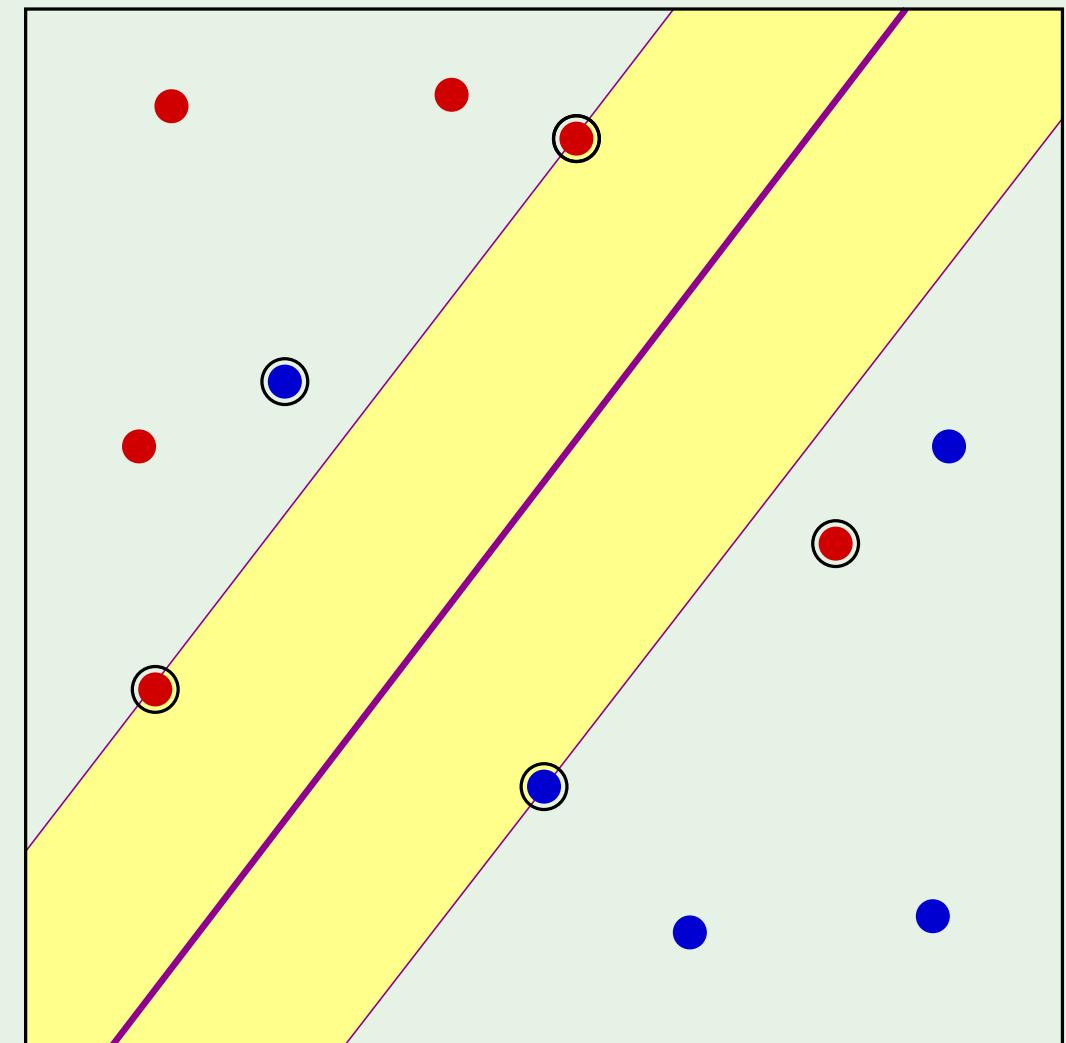
Types of support vectors

margin support vectors $(0 < \alpha_n < C)$

$$y_n (\mathbf{w}^\top \mathbf{x}_n + b) = 1 \quad (\xi_n = 0)$$

non-margin support vectors $(\alpha_n = C)$

$$y_n (\mathbf{w}^\top \mathbf{x}_n + b) < 1 \quad (\xi_n > 0)$$



Two technical observations

1. Hard margin: What if data is not linearly separable?

“primal \longrightarrow dual” breaks down

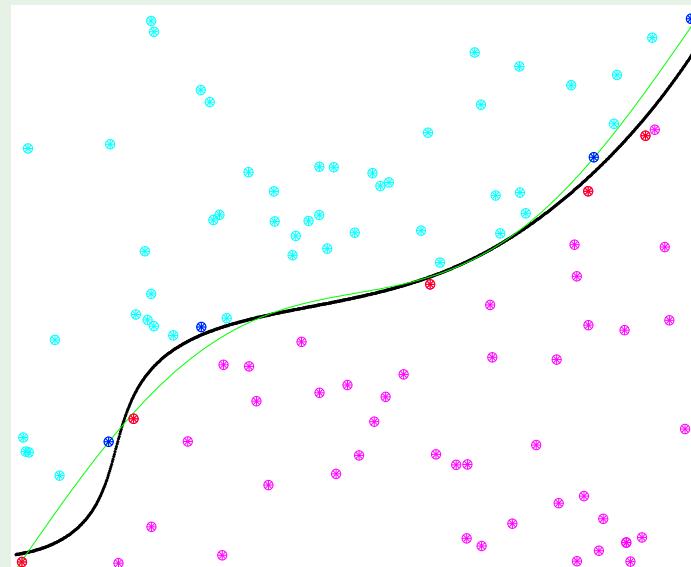
2. \mathcal{Z} : What if there is w_0 ?

All goes to b and $w_0 \rightarrow 0$

Review of Lecture 15

- Kernel methods

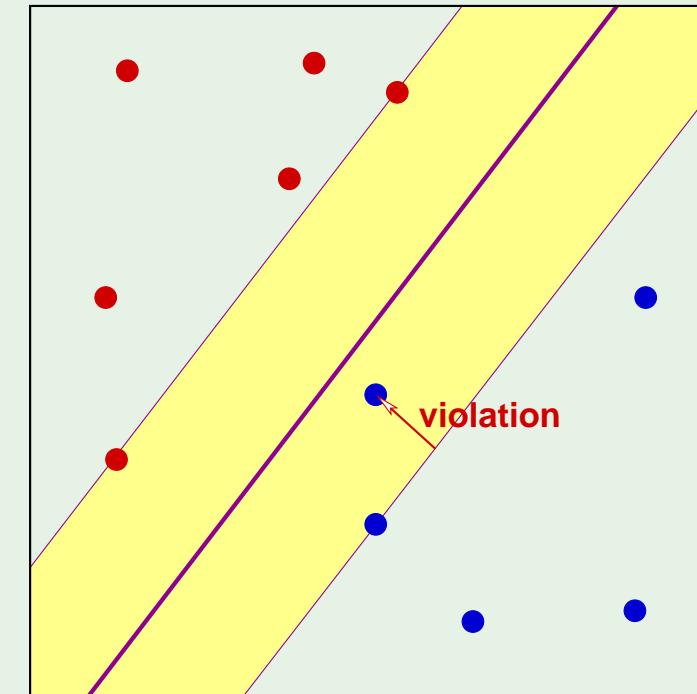
$$K(\mathbf{x}, \mathbf{x}') = \mathbf{z}^\top \mathbf{z}' \text{ for some } \mathcal{Z} \text{ space}$$



$$K(\mathbf{x}, \mathbf{x}') = \exp\left(-\gamma \|\mathbf{x} - \mathbf{x}'\|^2\right)$$

- Soft-margin SVM

$$\text{Minimize } \frac{1}{2} \mathbf{w}^\top \mathbf{w} + C \sum_{n=1}^N \xi_n$$

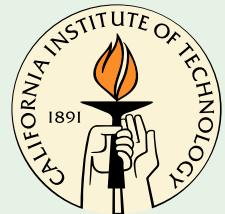


Same as hard margin, but $0 \leq \alpha_n \leq C$

Learning From Data

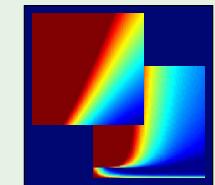
Yaser S. Abu-Mostafa
California Institute of Technology

Lecture 16: Radial Basis Functions



Sponsored by Caltech's Provost Office, E&AS Division, and IST

• Thursday, May 24, 2012



Outline

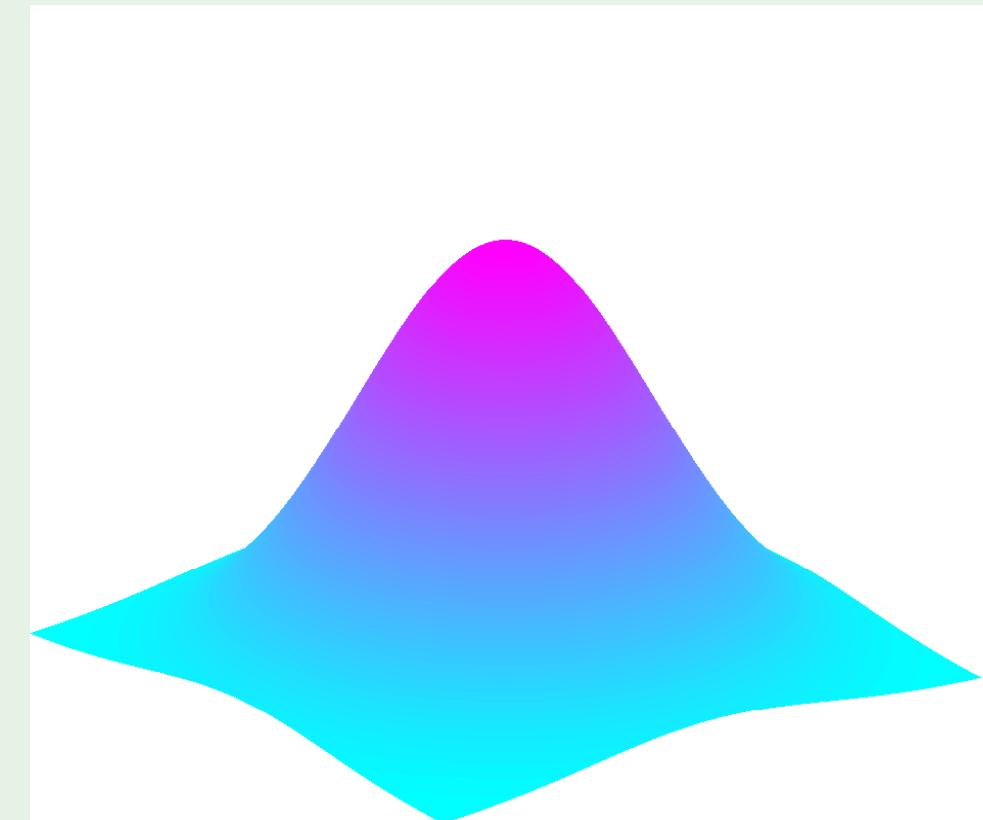
- RBF and nearest neighbors
- RBF and neural networks
- RBF and kernel methods
- RBF and regularization

Basic RBF model

Each $(\mathbf{x}_n, y_n) \in \mathcal{D}$ influences $h(\mathbf{x})$ based on $\underbrace{\|\mathbf{x} - \mathbf{x}_n\|}_{\text{radial}}$

Standard form:

$$h(\mathbf{x}) = \sum_{n=1}^N w_n \underbrace{\exp(-\gamma \|\mathbf{x} - \mathbf{x}_n\|^2)}_{\text{basis function}}$$



The learning algorithm

Finding w_1, \dots, w_N :

$$h(\mathbf{x}) = \sum_{n=1}^N w_n \exp\left(-\gamma \|\mathbf{x} - \mathbf{x}_n\|^2\right)$$

based on $\mathcal{D} = (\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)$

$E_{\text{in}} = 0$: $h(\mathbf{x}_n) = y_n$ for $n = 1, \dots, N$:

$$\sum_{m=1}^N w_m \exp\left(-\gamma \|\mathbf{x}_n - \mathbf{x}_m\|^2\right) = y_n$$

The solution

$$\sum_{m=1}^N \mathbf{w}_m \exp\left(-\gamma \|\mathbf{x}_n - \mathbf{x}_m\|^2\right) = y_n \quad N \text{ equations in } N \text{ unknowns}$$

$$\underbrace{\begin{bmatrix} \exp(-\gamma \|\mathbf{x}_1 - \mathbf{x}_1\|^2) & \dots & \exp(-\gamma \|\mathbf{x}_1 - \mathbf{x}_N\|^2) \\ \exp(-\gamma \|\mathbf{x}_2 - \mathbf{x}_1\|^2) & \dots & \exp(-\gamma \|\mathbf{x}_2 - \mathbf{x}_N\|^2) \\ \vdots & \vdots & \vdots \\ \exp(-\gamma \|\mathbf{x}_N - \mathbf{x}_1\|^2) & \dots & \exp(-\gamma \|\mathbf{x}_N - \mathbf{x}_N\|^2) \end{bmatrix}}_{\Phi} \underbrace{\begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_N \end{bmatrix}}_{\mathbf{w}} = \underbrace{\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}}_{\mathbf{y}}$$

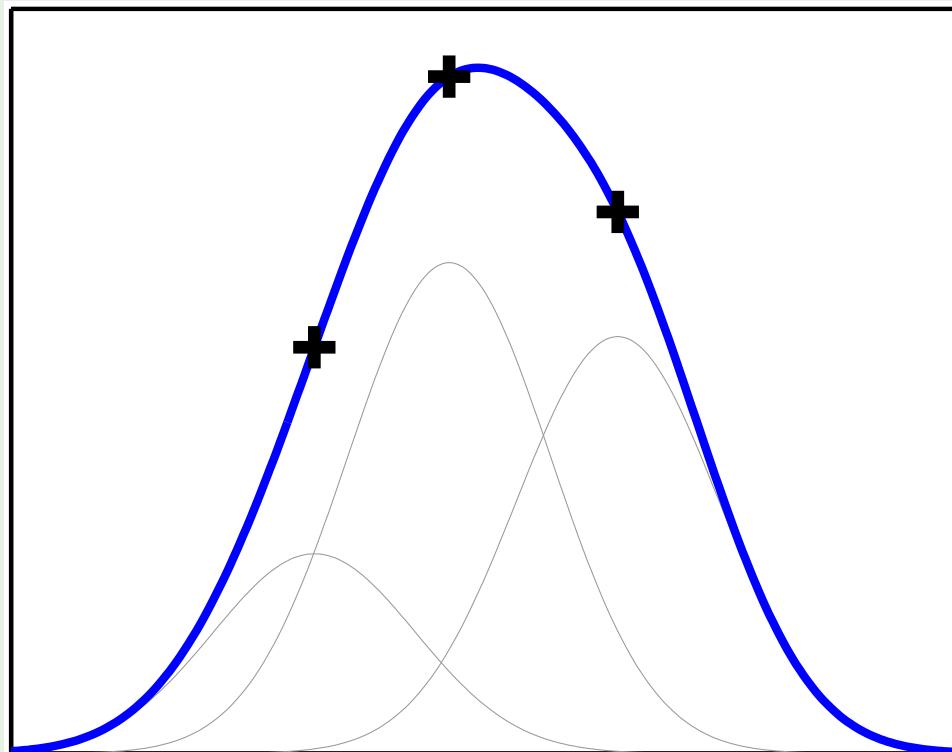
If Φ is invertible,

$$\mathbf{w} = \Phi^{-1}\mathbf{y}$$

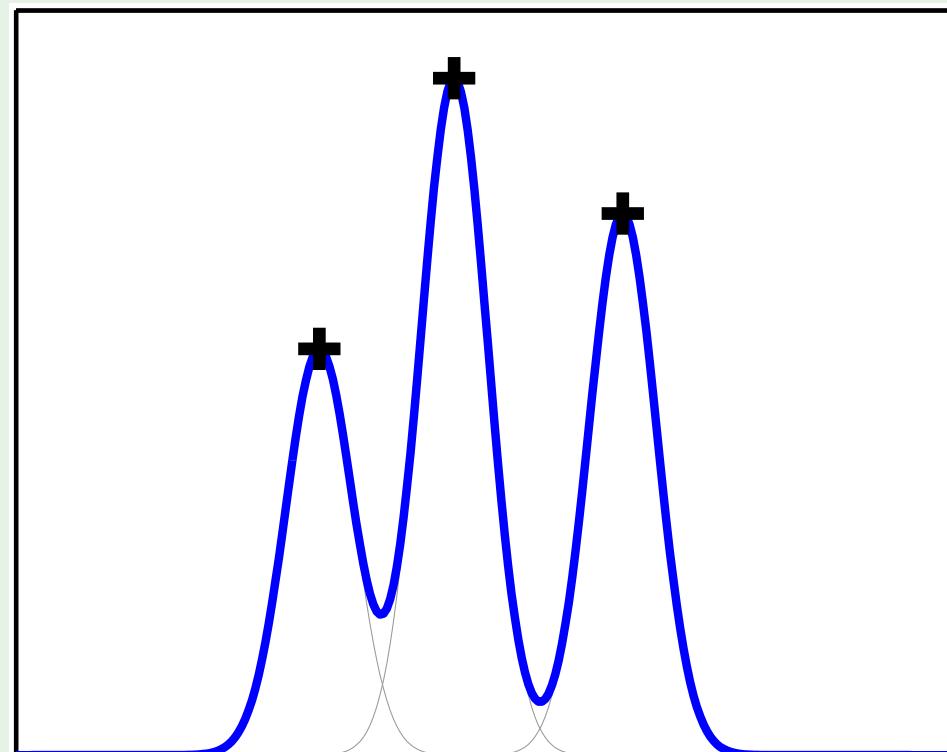
“exact interpolation”

The effect of γ

$$h(\mathbf{x}) = \sum_{n=1}^N w_n \exp\left(-\gamma \|\mathbf{x} - \mathbf{x}_n\|^2\right)$$



small γ



large γ

RBF for classification

$$h(\mathbf{x}) = \text{sign} \left(\sum_{n=1}^N w_n \exp \left(-\gamma \|\mathbf{x} - \mathbf{x}_n\|^2 \right) \right)$$

Learning: \sim linear regression for classification

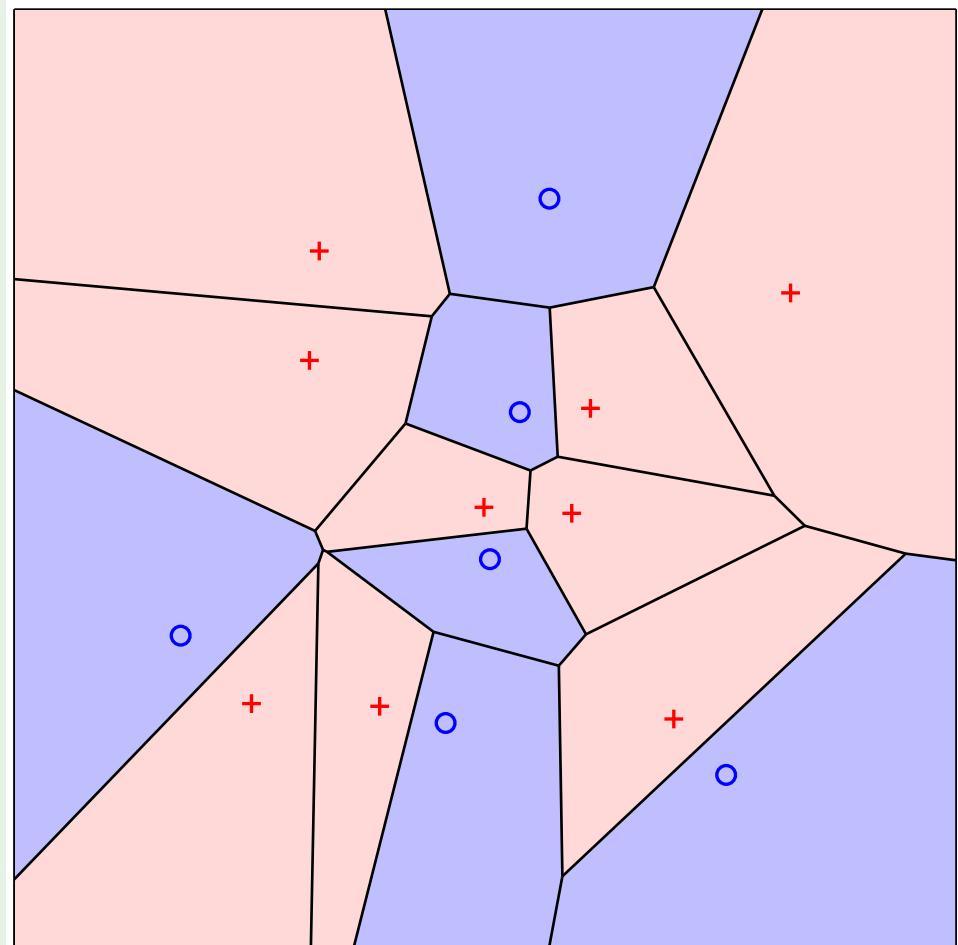
$$s = \sum_{n=1}^N w_n \exp \left(-\gamma \|\mathbf{x} - \mathbf{x}_n\|^2 \right)$$

Minimize $(s - y)^2$ on \mathcal{D} $y = \pm 1$

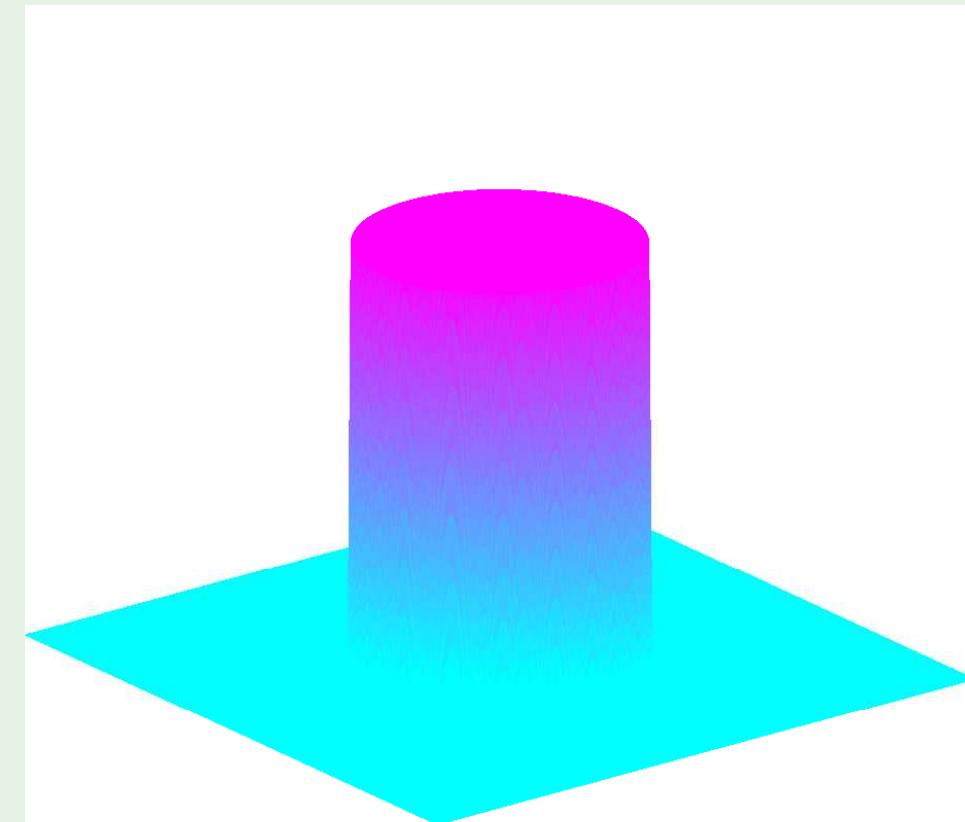
$$h(\mathbf{x}) = \text{sign}(s)$$

Relationship to nearest-neighbor method

Adopt the y value of a nearby point:



similar effect by a basis function:



RBF with K centers

N parameters w_1, \dots, w_N based on N data points

Use $K \ll N$ centers: μ_1, \dots, μ_K instead of $\mathbf{x}_1, \dots, \mathbf{x}_N$

$$h(\mathbf{x}) = \sum_{k=1}^K w_k \exp\left(-\gamma \|\mathbf{x} - \mu_k\|^2\right)$$

1. How to choose the centers μ_k
2. How to choose the weights w_k

Choosing the centers

Minimize the distance between \mathbf{x}_n and the **closest** center $\boldsymbol{\mu}_k$:

K -means clustering

Split $\mathbf{x}_1, \dots, \mathbf{x}_N$ into clusters S_1, \dots, S_K

$$\text{Minimize} \sum_{k=1}^K \sum_{\mathbf{x}_n \in S_k} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2$$

Unsupervised learning ☺

NP-hard ☹

An iterative algorithm

Lloyd's algorithm: Iteratively minimize

$$\sum_{k=1}^K \sum_{\mathbf{x}_n \in S_k} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2 \quad \text{w.r.t. } \boldsymbol{\mu}_k, S_k$$

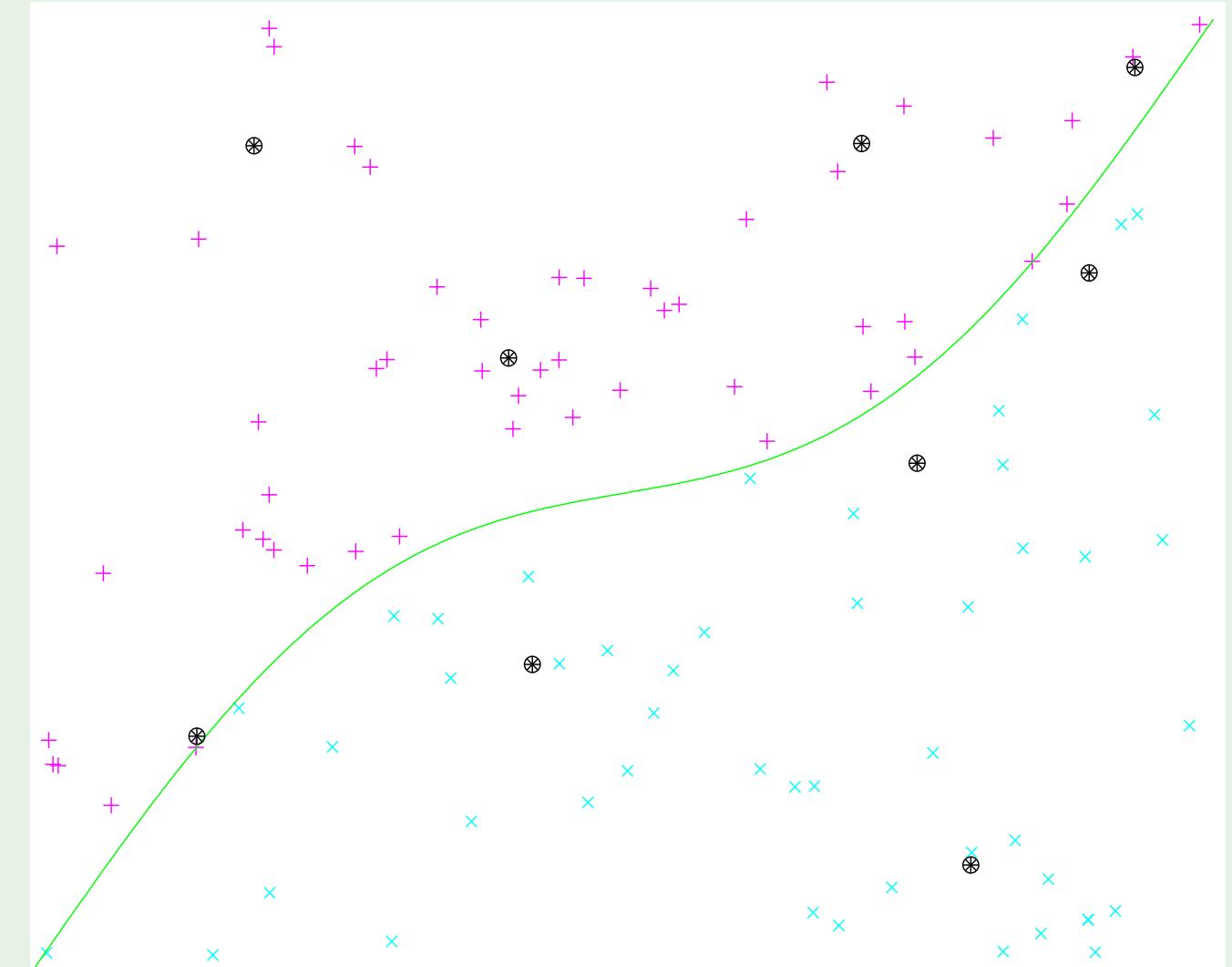
$$\boldsymbol{\mu}_k \leftarrow \frac{1}{|S_k|} \sum_{\mathbf{x}_n \in S_k} \mathbf{x}_n$$

$$S_k \leftarrow \{\mathbf{x}_n : \|\mathbf{x}_n - \boldsymbol{\mu}_k\| \leq \text{all } \|\mathbf{x}_n - \boldsymbol{\mu}_\ell\|\}$$

Convergence \longrightarrow local minimum

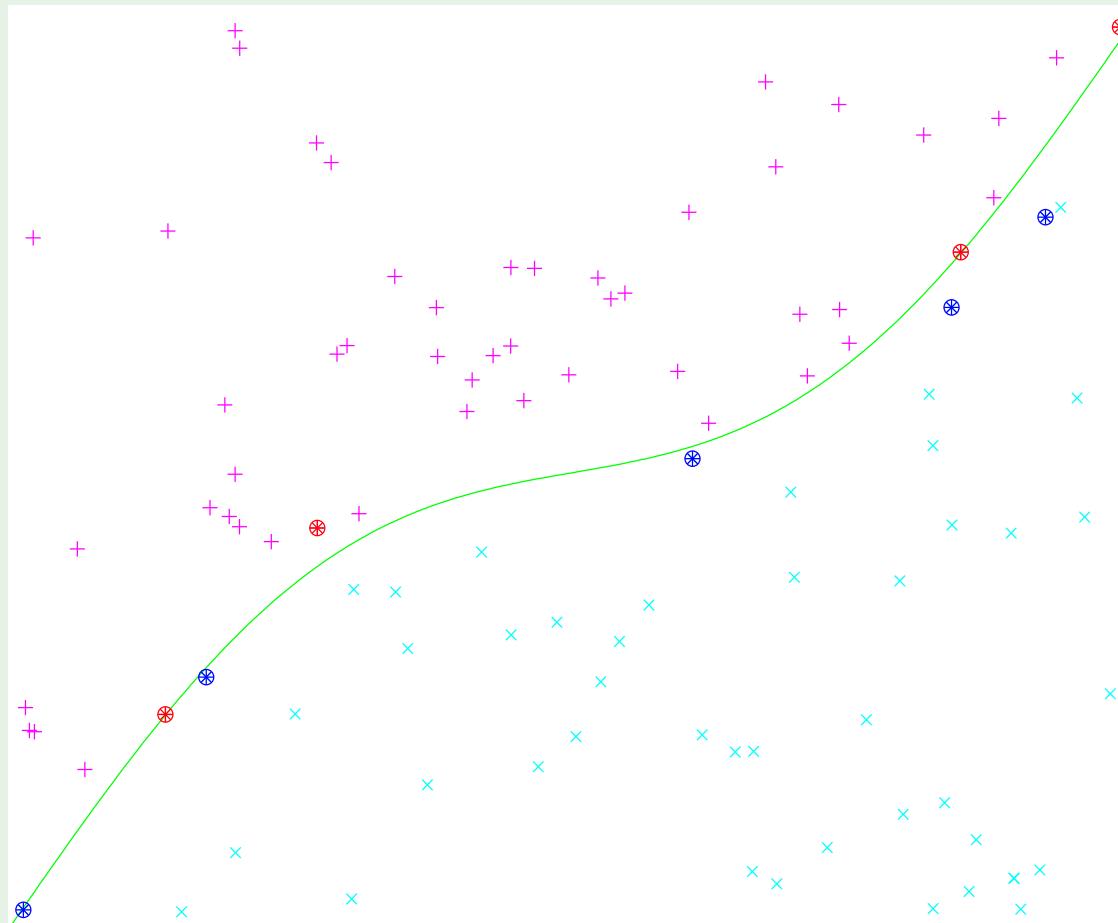
Lloyd's algorithm in action

1. Get the data points
2. Only the inputs!
3. Initialize the centers
4. Iterate
5. These are your μ_k 's

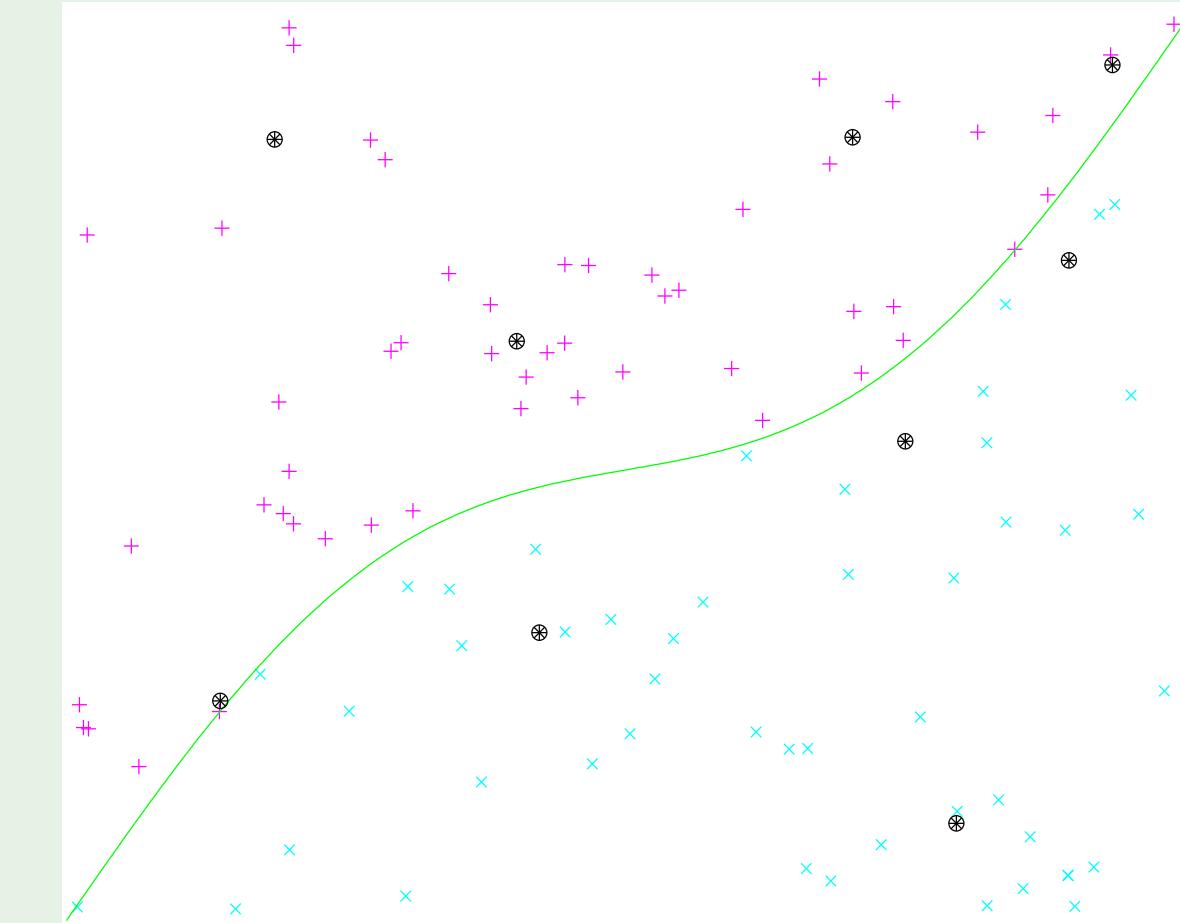


Centers versus support vectors

support vectors



RBF centers



Choosing the weights

$$\sum_{k=1}^K w_k \exp\left(-\gamma \|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2\right) \approx y_n \quad N \text{ equations in } K < N \text{ unknowns}$$

$$\begin{bmatrix} \exp(-\gamma \|\mathbf{x}_1 - \boldsymbol{\mu}_1\|^2) & \dots & \exp(-\gamma \|\mathbf{x}_1 - \boldsymbol{\mu}_K\|^2) \\ \exp(-\gamma \|\mathbf{x}_2 - \boldsymbol{\mu}_1\|^2) & \dots & \exp(-\gamma \|\mathbf{x}_2 - \boldsymbol{\mu}_K\|^2) \\ \vdots & \vdots & \vdots \\ \exp(-\gamma \|\mathbf{x}_N - \boldsymbol{\mu}_1\|^2) & \dots & \exp(-\gamma \|\mathbf{x}_N - \boldsymbol{\mu}_K\|^2) \end{bmatrix} \underbrace{\begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_K \end{bmatrix}}_{\mathbf{w}} \approx \underbrace{\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}}_{\mathbf{y}}$$

If $\Phi^\top \Phi$ is invertible,

$$\boxed{\mathbf{w} = (\Phi^\top \Phi)^{-1} \Phi^\top \mathbf{y}}$$

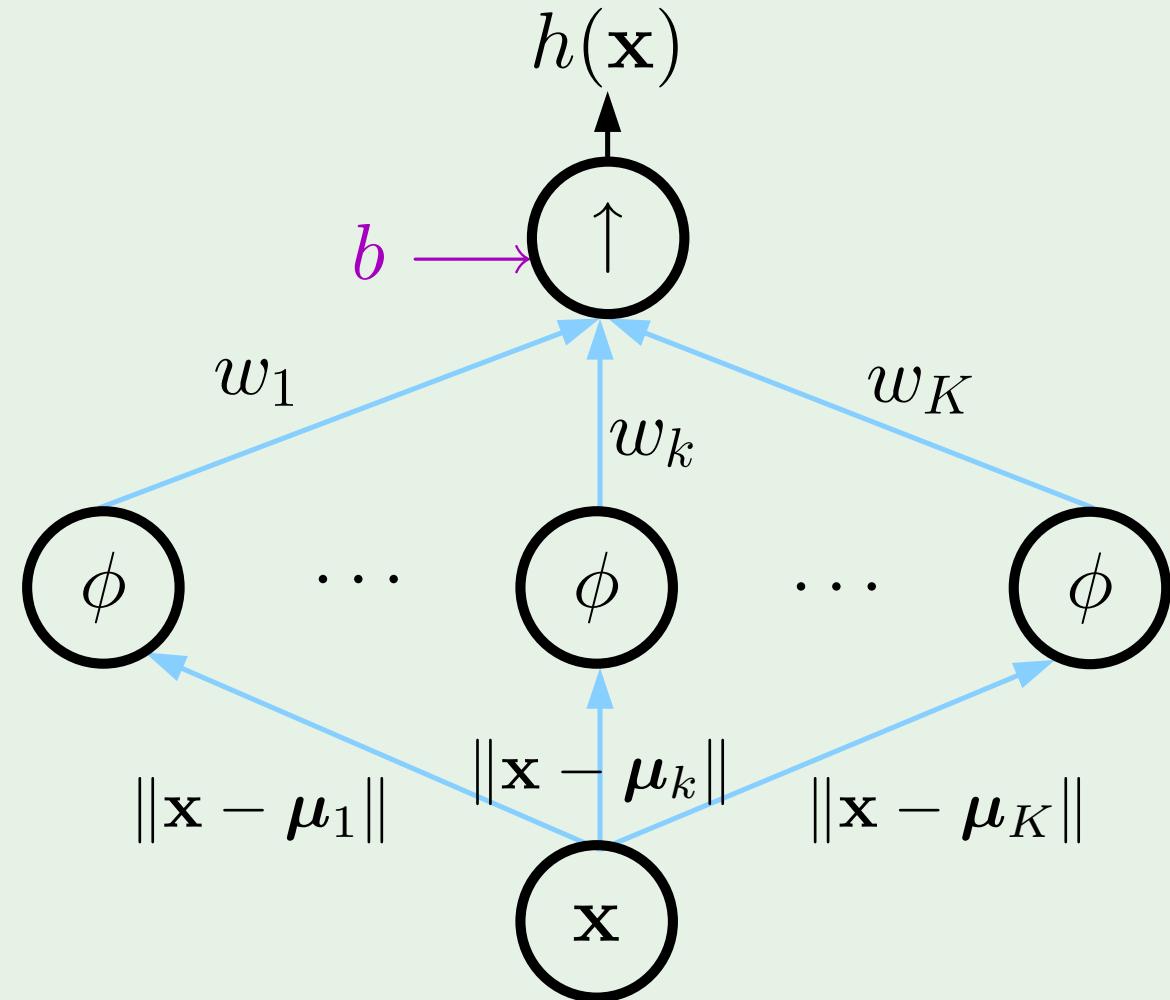
pseudo-inverse

RBF network

The “features” are $\exp(-\gamma \|\mathbf{x} - \boldsymbol{\mu}_k\|^2)$

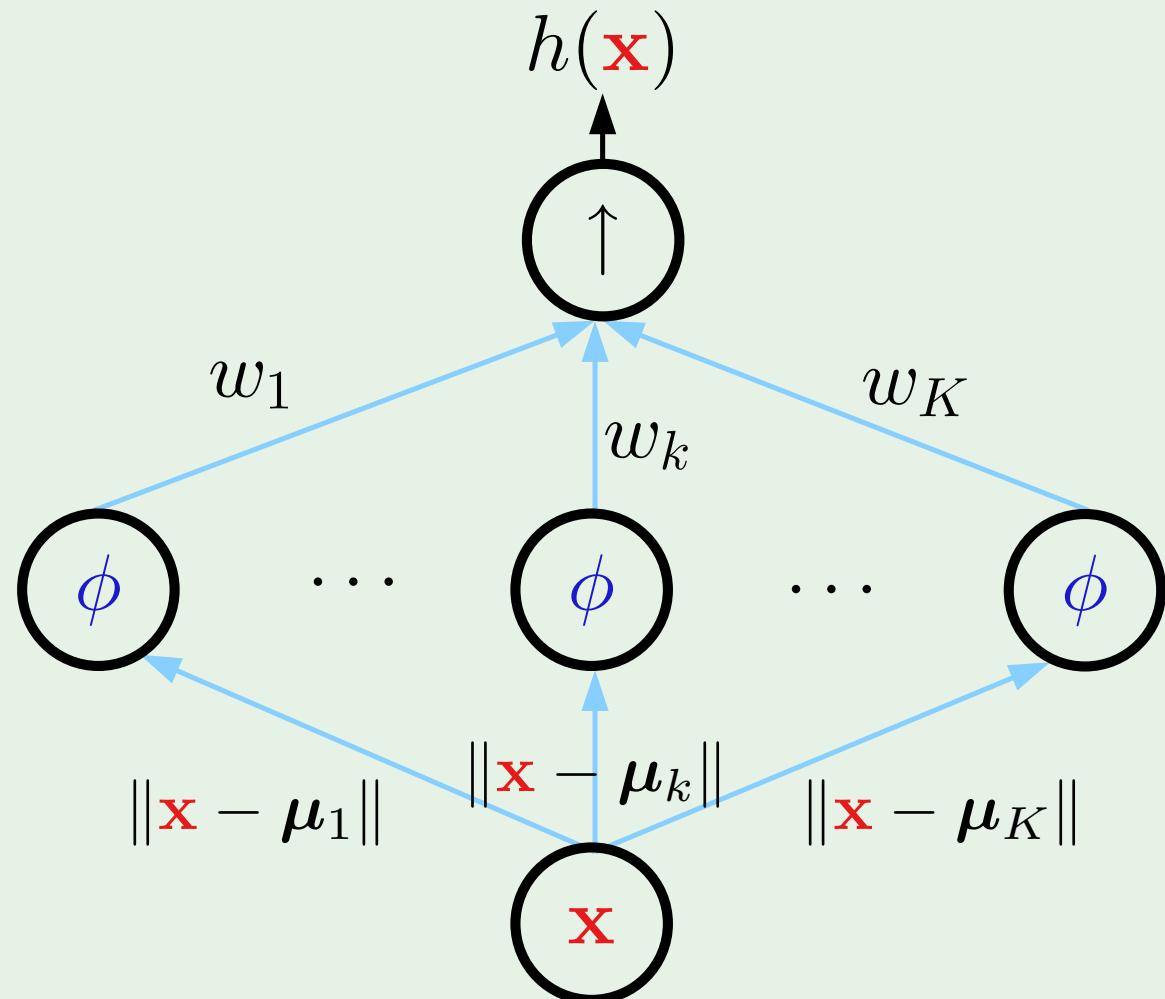
Nonlinear transform depends on \mathcal{D}

\implies No longer a linear model

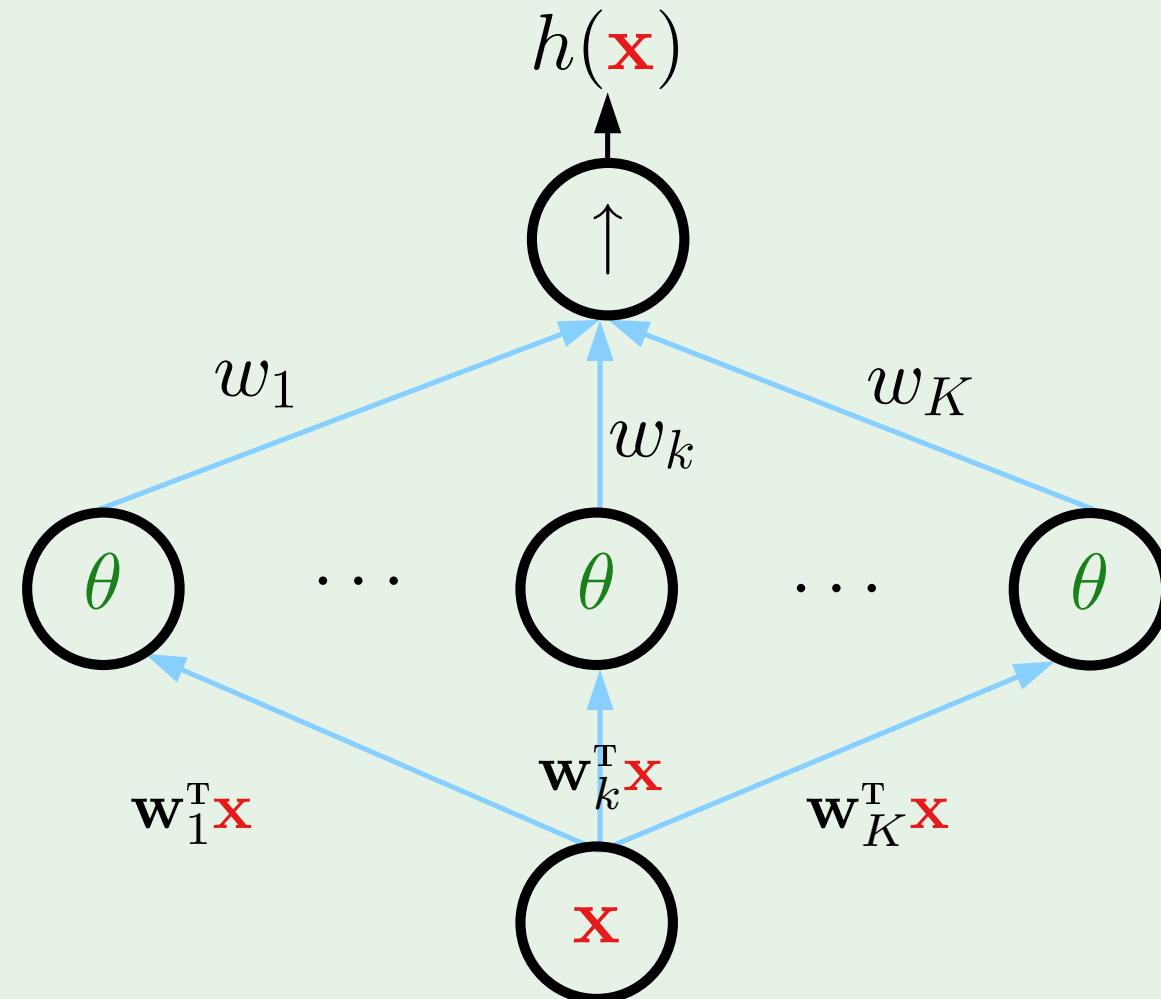


A bias term (b or w_0) is often added

Compare to neural networks



RBF network



neural network

Choosing γ

Treating γ as a parameter to be learned
$$h(\mathbf{x}) = \sum_{k=1}^K w_k \exp\left(-\gamma \|\mathbf{x} - \boldsymbol{\mu}_k\|^2\right)$$

Iterative approach (\sim EM algorithm in mixture of Gaussians):

1. Fix γ , solve for w_1, \dots, w_K
2. Fix w_1, \dots, w_K , minimize error w.r.t. γ

We can have a different γ_k for each center $\boldsymbol{\mu}_k$

Outline

- RBF and nearest neighbors
- RBF and neural networks
- RBF and kernel methods
- RBF and regularization

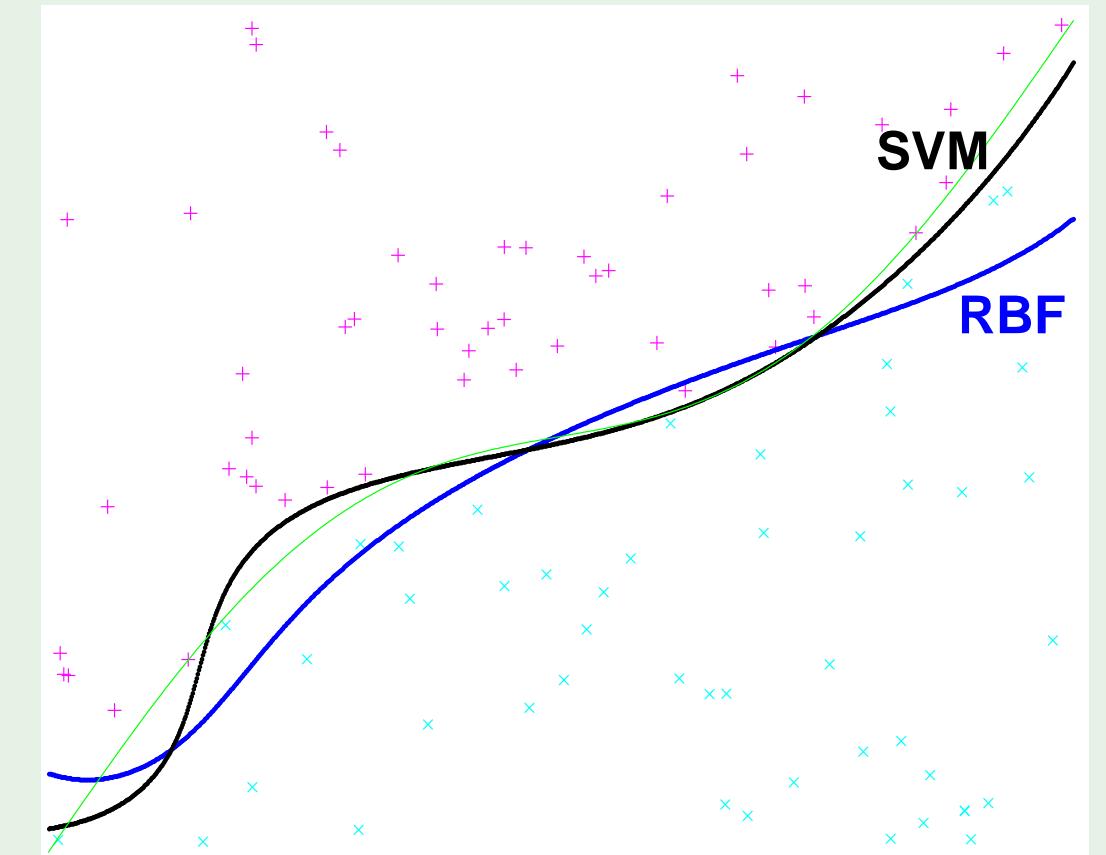
RBF versus its SVM kernel

SVM kernel implements:

$$\text{sign} \left(\sum_{\alpha_n > 0} \alpha_n y_n \exp \left(-\gamma \| \mathbf{x} - \mathbf{x}_n \|^2 \right) + b \right)$$

Straight RBF implements:

$$\text{sign} \left(\sum_{k=1}^K w_k \exp \left(-\gamma \| \mathbf{x} - \boldsymbol{\mu}_k \|^2 \right) + b \right)$$



RBF and regularization

RBF can be derived based purely on regularization:

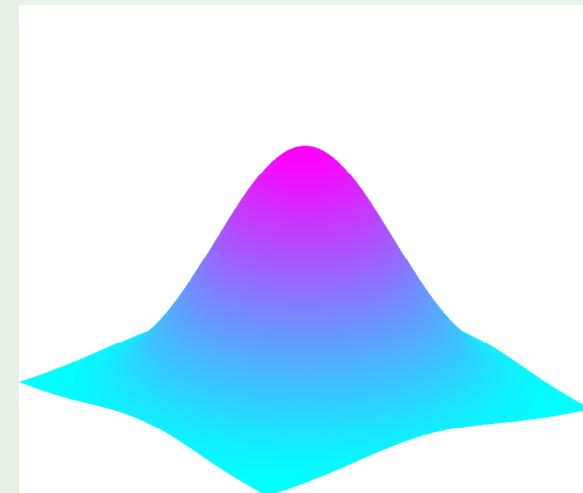
$$\sum_{n=1}^N (h(x_n) - y_n)^2 + \lambda \sum_{k=0}^{\infty} a_k \int_{-\infty}^{\infty} \left(\frac{d^k h}{dx^k} \right)^2 dx$$

“smoothest interpolation”

Review of Lecture 16

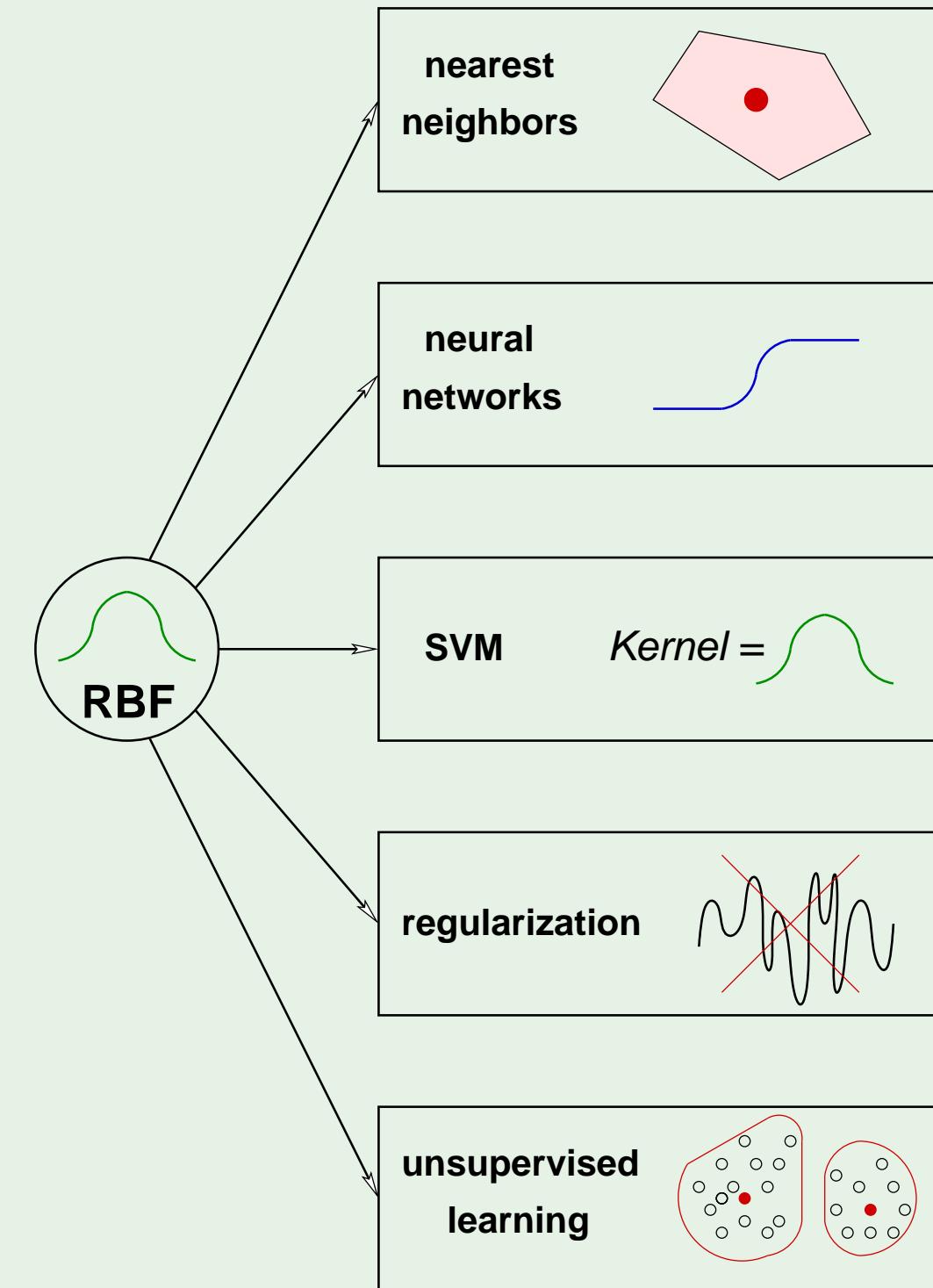
- Radial Basis Functions

$$h(\mathbf{x}) = \sum_{k=1}^K w_k \exp\left(-\gamma \|\mathbf{x} - \boldsymbol{\mu}_k\|^2\right)$$



Choose $\boldsymbol{\mu}_k$'s: Lloyd's algorithm

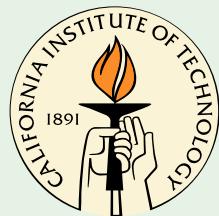
Choose w_k 's: Pseudo-inverse



Learning From Data

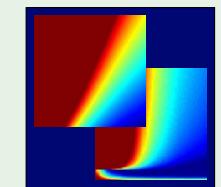
Yaser S. Abu-Mostafa
California Institute of Technology

Lecture 17: Three Learning Principles



Sponsored by Caltech's Provost Office, E&AS Division, and IST

• Tuesday, May 29, 2012



Outline

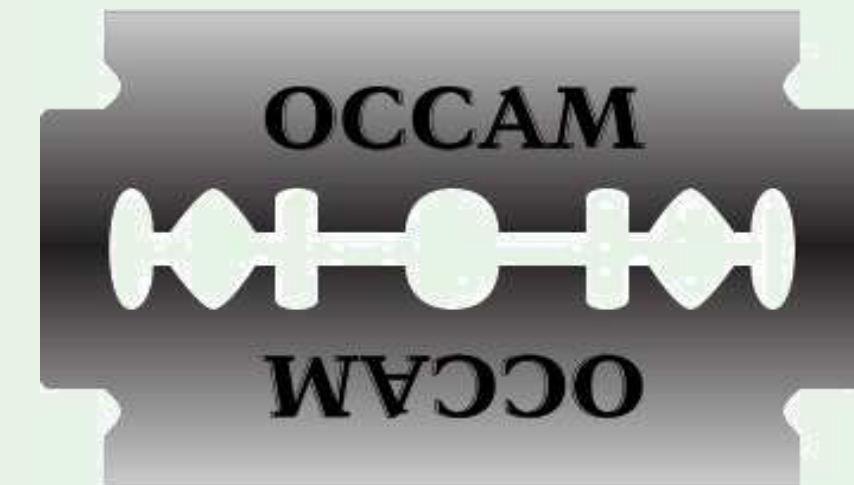
- Occam's Razor
- Sampling Bias
- Data Snooping

Recurring theme - simple hypotheses

A “quote” by Einstein:

An explanation of the data should be made *as simple as possible, but no simpler*

The razor: symbolic of a principle set by William of Occam



Occam's Razor

The simplest model that fits the data is also the most plausible.

Two questions:

1. What does it mean for a model to be simple?
2. How do we know that simpler is better?

First question: ‘simple’ means?

Measures of complexity - two types: **complexity of h** and **complexity of \mathcal{H}**

Complexity of h : MDL, order of a polynomial

Complexity of \mathcal{H} : Entropy, VC dimension

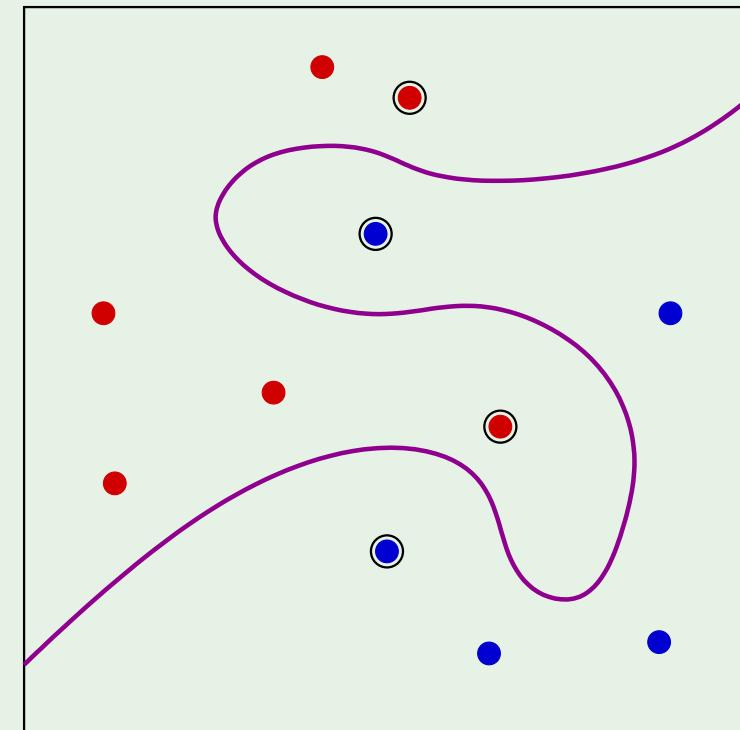
- When we think of simple, it's in terms of h
- Proofs use simple in terms of \mathcal{H}

and the link is ...

counting: ℓ bits specify h $\implies h$ is one of 2^ℓ elements of a set \mathcal{H}

Real-valued parameters? Example: 17th order polynomial - complex and one of "many"

Exceptions? Looks complex but is one of few - SVM



Puzzle 1: Football oracle

0000000000000000**1111111111111111**
00000000**11111111**0000000011111111
000011110000**1111**0000111100001111
0011001100**11**00110011001100110011
0101010101**01**01010101010101010101



0
1
0
1
1

- Letter predicting game outcome
- Good call!
- More letters - for 5 weeks
- Perfect record!
- Want more? \$50 charge ☺
- Should you pay?

Second question: Why is simpler better?

Better doesn't mean more elegant! It means better out-of-sample performance

The basic argument: (formal proof under different idealized conditions)

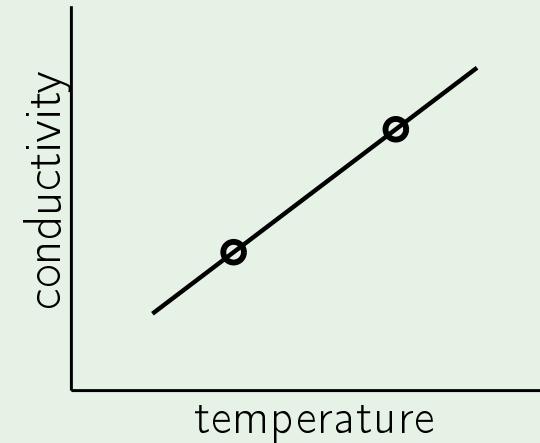
Fewer simple hypotheses than complex ones
 $m_{\mathcal{H}}(N)$

⇒ less likely to fit a given data set
 $m_{\mathcal{H}}(N)/2^N$

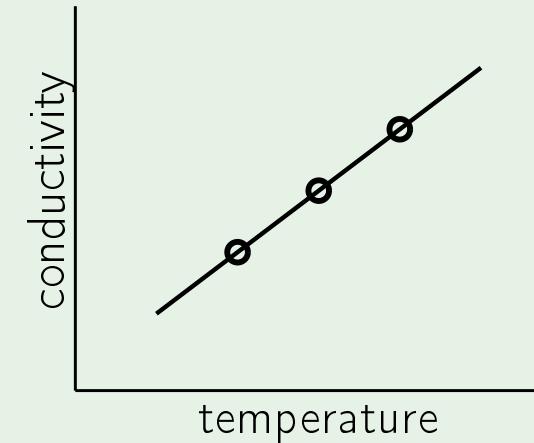
⇒ more significant when it happens

The postal scam: $m_{\mathcal{H}}(N) = 1$ versus 2^N

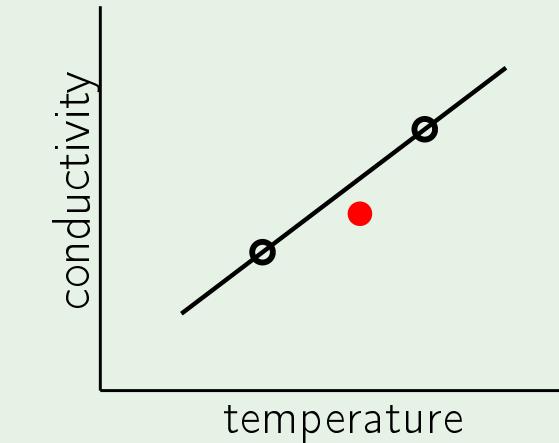
A fit that means nothing



Scientist A



Scientist B



"falsifiable"

Conductivity linear in temperature?

Two scientists conduct experiments

What evidence do A and B provide?

Outline

- Occam's Razor
- Sampling Bias
- Data Snooping

Puzzle 2: Presidential election

In 1948, Truman ran against Dewey in close elections

A newspaper ran a phone poll of how people voted

Dewey won the poll decisively - newspaper declared:



On to the victory rally ...

... of Truman ☺

It's not δ 's fault:

$$\mathbb{P} [|E_{\text{in}} - E_{\text{out}}| > \epsilon] \leq \delta$$



The bias

In 1948, phones were expensive.

If the data is sampled in a biased way, learning will produce a similarly biased outcome.

Example: normal period in the market

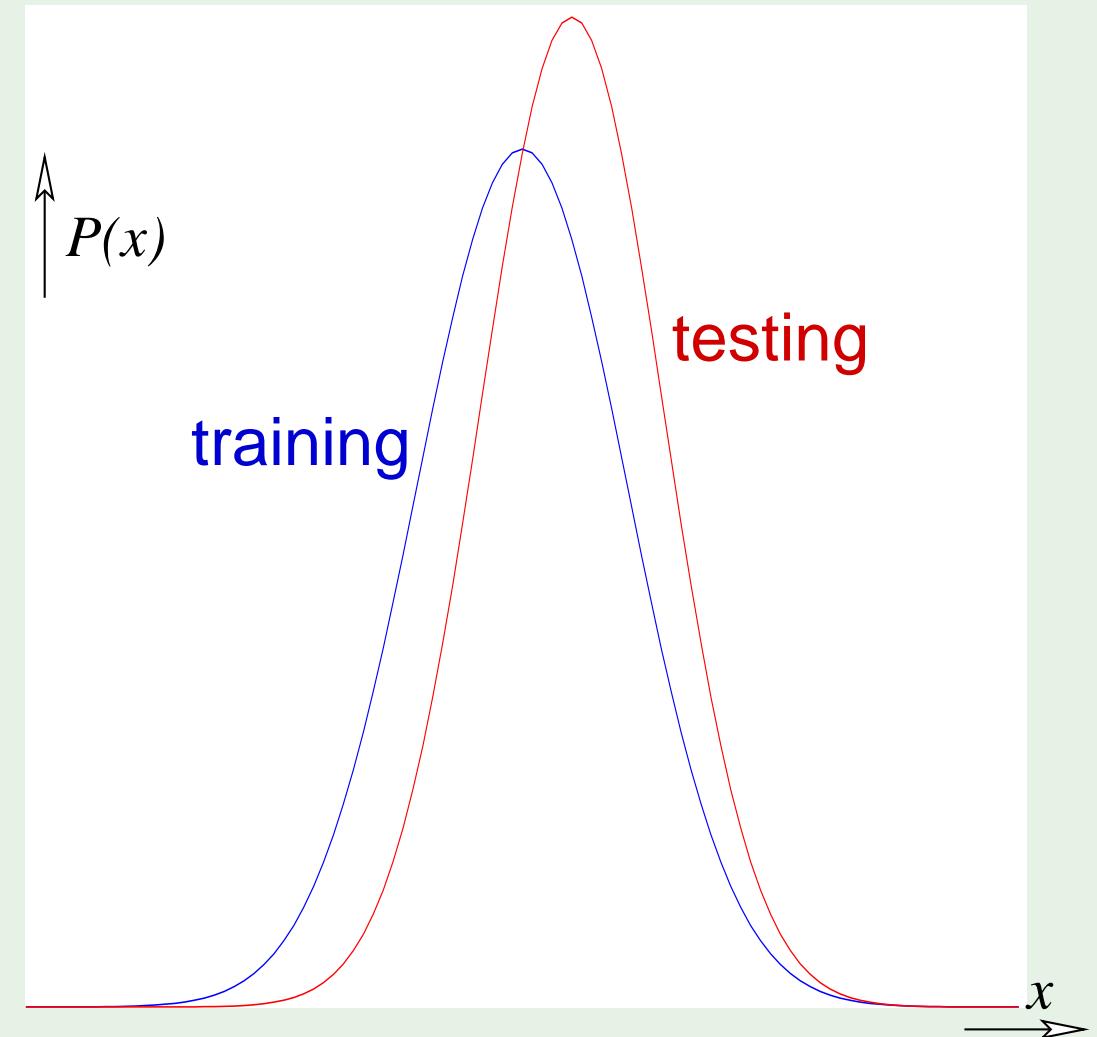
Testing: live trading in real market

Matching the distributions

Methods to match training and testing distributions

Doesn't work if:

Region has $P = 0$ in training, but $P > 0$ in testing



Puzzle 3: Credit approval

Historical records of customers

Input: information on credit application:

Target: profitable for the bank

age	23 years
gender	male
annual salary	\$30,000
years in residence	1 year
years in job	1 year
current debt	\$15,000
...	...

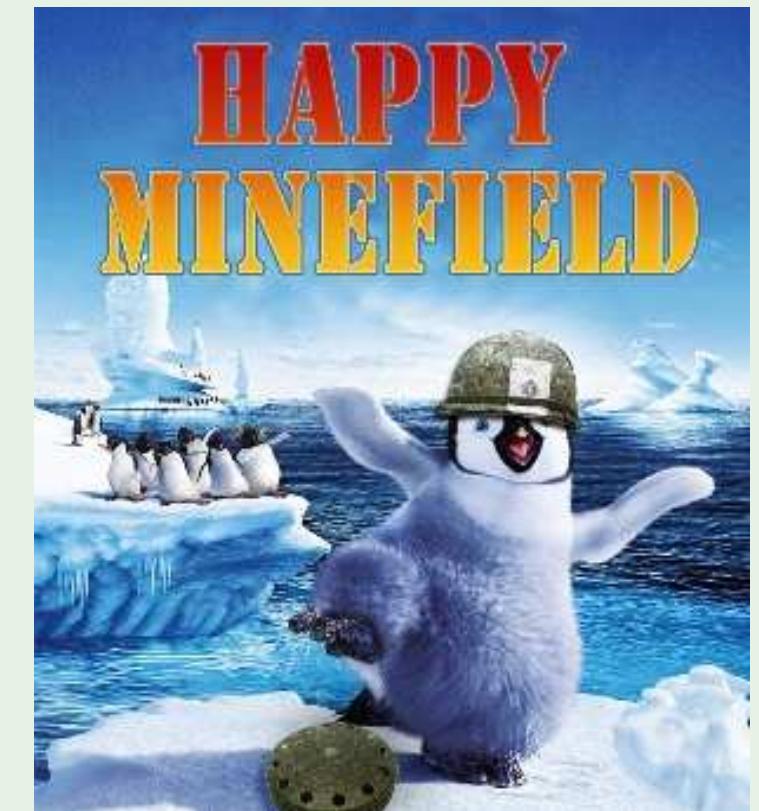
Outline

- Occam's Razor
- Sampling Bias
- Data Snooping

The principle

If a data set has affected any step in the learning process,
its ability to assess the outcome has been compromised.

Most common trap for practitioners - many ways to slip ☹



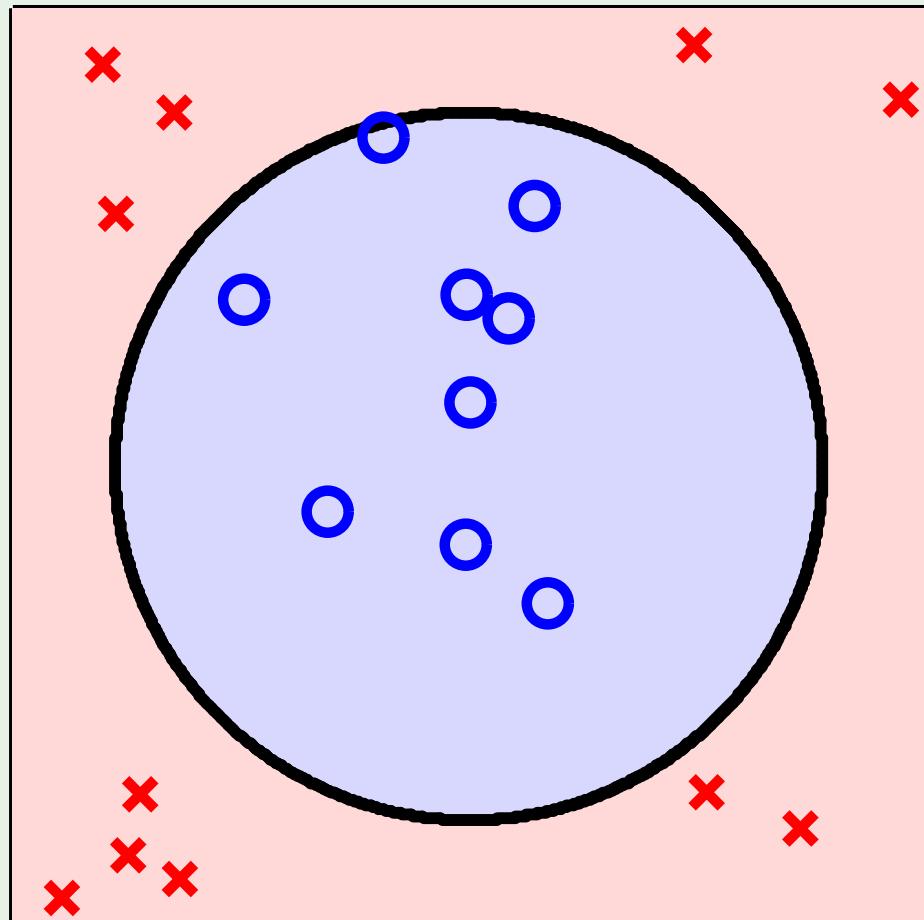
Looking at the data

Remember nonlinear transforms?

$$\mathbf{z} = (1, x_1, x_2, x_1x_2, x_1^2, x_2^2)$$

or $\mathbf{z} = (1, x_1^2, x_2^2)$ or $\mathbf{z} = (1, x_1^2 + x_2^2)$

Snooping involves \mathcal{D} , not other information

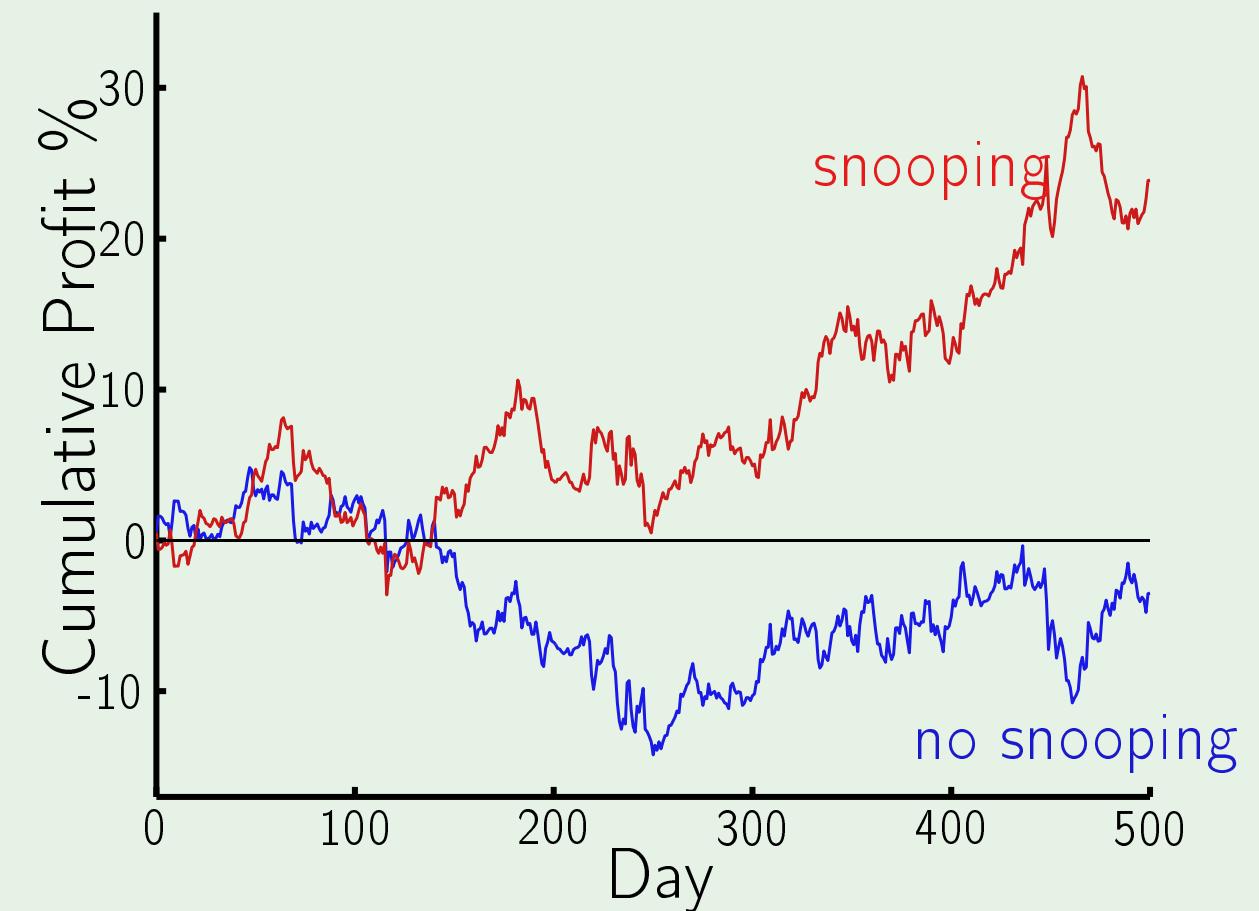


Puzzle 4: Financial forecasting

Predict US Dollar versus British Pound

Normalize data, split randomly: $\mathcal{D}_{\text{train}}, \mathcal{D}_{\text{test}}$

Train on $\mathcal{D}_{\text{train}}$ only, test g on $\mathcal{D}_{\text{test}}$



$$\Delta r_{-20}, \Delta r_{-19}, \dots, \Delta r_{-1} \rightarrow \Delta r_0$$

Reuse of a data set

Trying one model after the other **on the same data set**, you will eventually ‘succeed’

If you torture the data long enough, it will confess

VC dimension of the **total** learning model

May include what **others** tried!

Key problem: matching a **particular** data set

Two remedies

1. **Avoid** data snooping
strict discipline
2. **Account for** data snooping
how much data contamination

Puzzle 5: Bias via snooping

Testing long-term performance of "buy and hold" in stocks. Use **50 years** worth of data

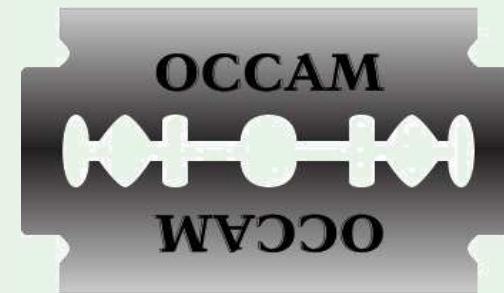
- All currently traded companies in S&P500
- Assume you strictly followed buy and hold
- Would have made great profit!

Sampling bias caused by 'snooping'

Review of Lecture 17

- Occam's Razor

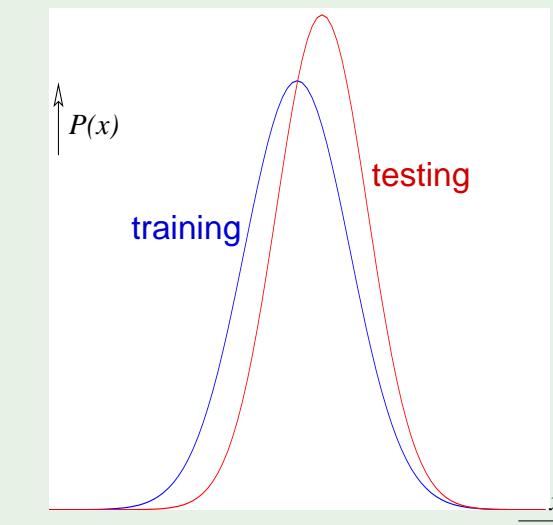
The simplest model that fits the data is also the most plausible.



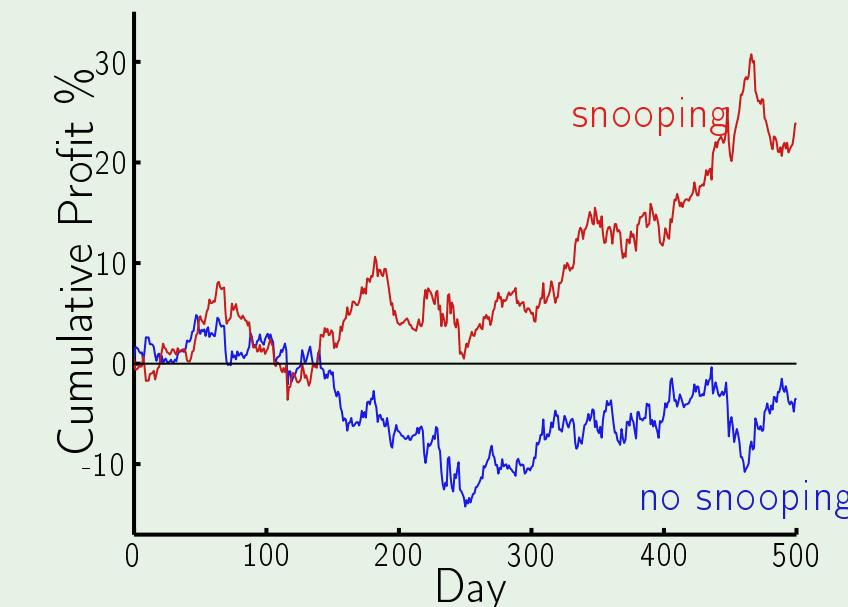
complexity of h \longleftrightarrow complexity of \mathcal{H}

unlikely event \longleftrightarrow significant if it happens

- Sampling bias



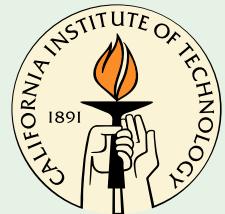
- Data snooping



Learning From Data

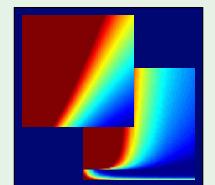
Yaser S. Abu-Mostafa
California Institute of Technology

Lecture 18: Epilogue



Sponsored by Caltech's Provost Office, E&AS Division, and IST

• Thursday, May 31, 2012



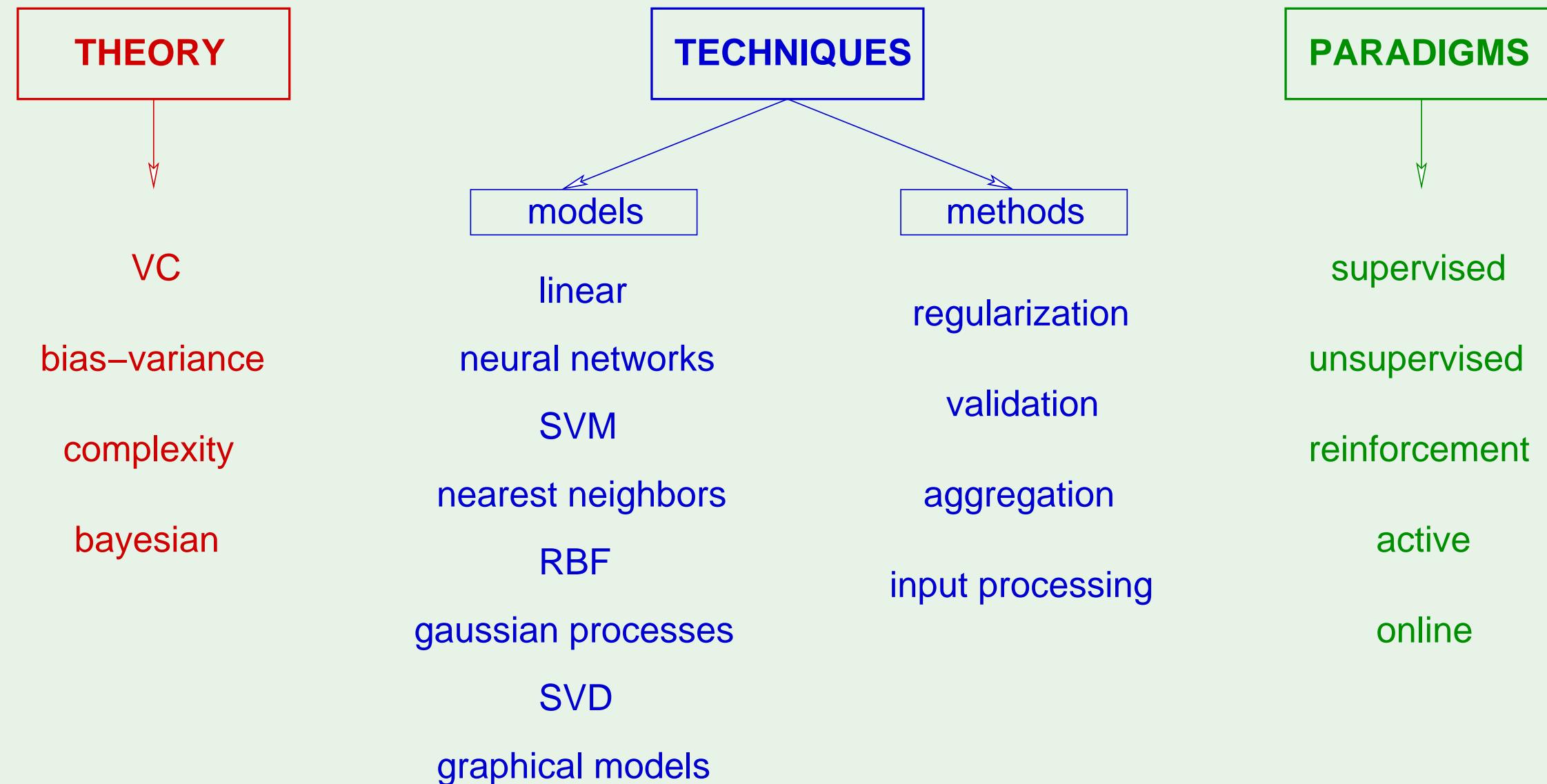
Outline

- The map of machine learning
- Bayesian learning
- Aggregation methods
- Acknowledgments

It's a jungle out there

semi-supervised learning	overfitting	stochastic gradient descent	SVM	<i>Q</i> learning
Gaussian processes	deterministic noise	data snooping		learning curves
<i>distribution-free</i>	<i>linear regression</i>	VC dimension		mixture of experts
collaborative filtering	nonlinear transformation	sampling bias	<i>neural networks</i>	<i>no free lunch</i>
decision trees	<i>RBF</i>	<i>training versus testing</i>	noisy targets	Bayesian prior
active learning	linear models	bias-variance tradeoff		weak learners
<i>ordinal regression</i>	cross validation	logistic regression	<i>data contamination</i>	hidden Markov models
ensemble learning	error measures	types of learning	perceptrons	graphical models
exploration versus exploitation	is learning feasible?	<i>kernel methods</i>		Boltzmann machines
clustering	regularization	weight decay	<i>Occam's razor</i>	

The map



Outline

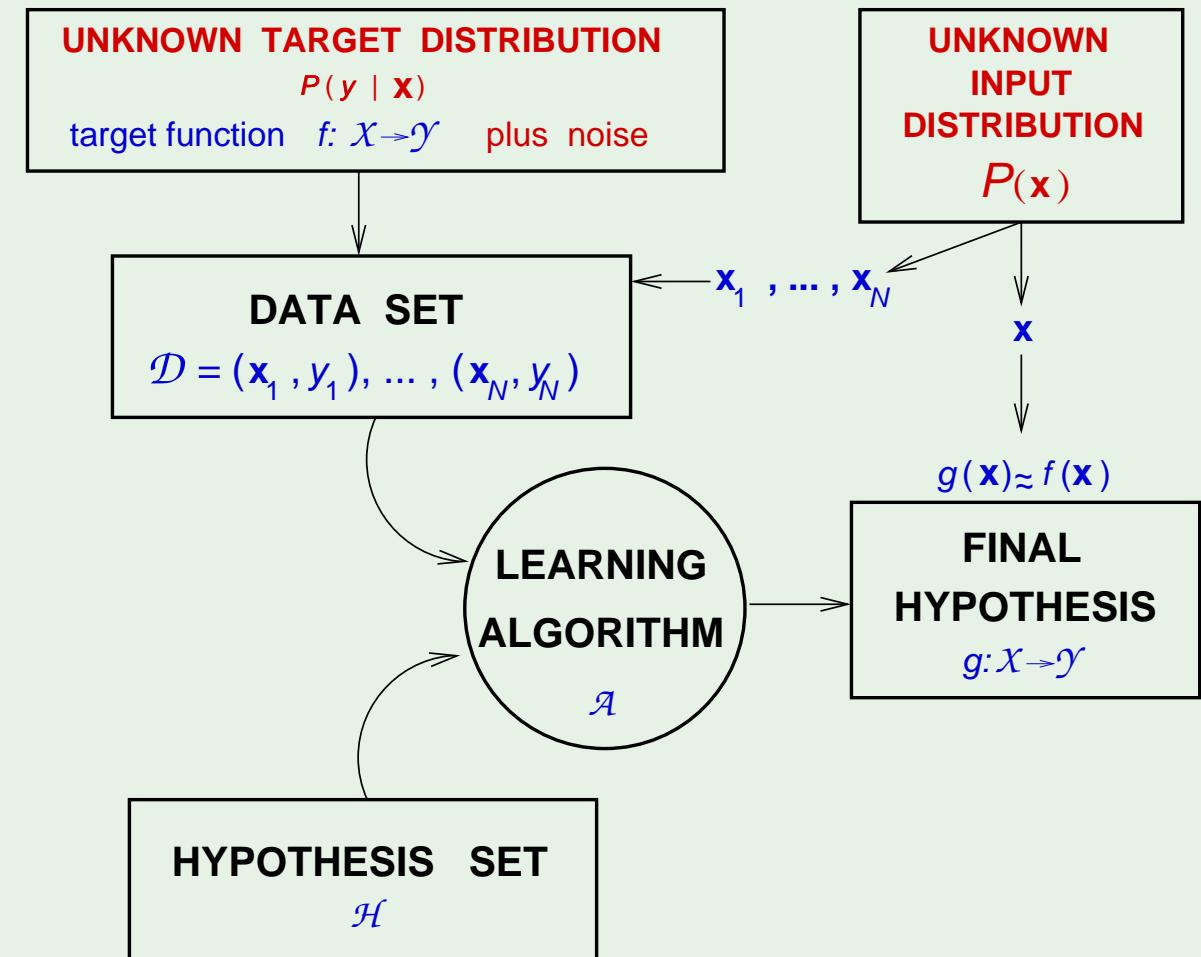
- The map of machine learning
- Bayesian learning
- Aggregation methods
- Acknowledgments

Probabilistic approach

Extend probabilistic role to all components

$P(\mathcal{D} | h = f)$ decides which h (likelihood)

How about $P(h = f | \mathcal{D})$?



The prior

$P(\textcolor{violet}{h} = f \mid \mathcal{D})$ requires an additional probability distribution:

$$P(\textcolor{violet}{h} = f \mid \mathcal{D}) = \frac{P(\mathcal{D} \mid \textcolor{violet}{h} = f) P(\textcolor{violet}{h} = f)}{P(\mathcal{D})} \propto P(\mathcal{D} \mid \textcolor{violet}{h} = f) P(\textcolor{violet}{h} = f)$$

$P(\textcolor{violet}{h} = f)$ is the **prior**

$P(\textcolor{violet}{h} = f \mid \mathcal{D})$ is the **posterior**

Given the prior, we have the full distribution

Example of a prior

Consider a perceptron: $\textcolor{violet}{h}$ is determined by $\mathbf{w} = w_0, w_1, \dots, w_d$

A possible prior on \mathbf{w} : Each w_i is independent, uniform over $[-1, 1]$

This determines the prior over $\textcolor{violet}{h}$ - $P(\textcolor{violet}{h} = f)$

Given \mathcal{D} , we can compute $P(\mathcal{D} \mid \textcolor{violet}{h} = f)$

Putting them together, we get $P(\textcolor{violet}{h} = f \mid \mathcal{D})$

$$\propto P(\textcolor{violet}{h} = f)P(\mathcal{D} \mid \textcolor{violet}{h} = f)$$

A prior is an assumption

Even the most “neutral” prior:



The true equivalent would be:



If we knew the prior

... we could compute $P(\mathbf{h} = f \mid \mathcal{D})$ for every $\mathbf{h} \in \mathcal{H}$

\implies we can find the most probable \mathbf{h} given the data

we can derive $\mathbb{E}(\mathbf{h}(\mathbf{x}))$ for every \mathbf{x}

we can derive the **error bar** for every \mathbf{x}

we can derive everything in a principled way

When is Bayesian learning justified?

1. The prior is **valid**

trumps all other methods

2. The prior is **irrelevant**

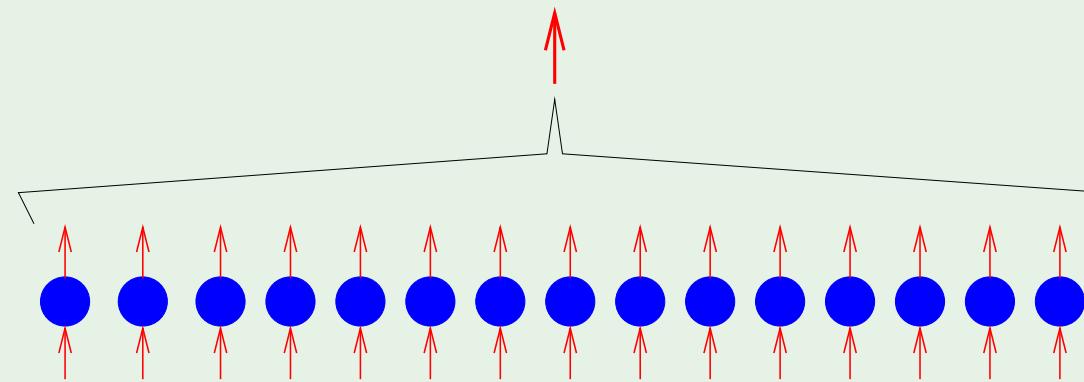
just a computational catalyst

Outline

- The map of machine learning
- Bayesian learning
- Aggregation methods
- Acknowledgments

What is aggregation?

Combining different solutions h_1, h_2, \dots, h_T that were trained on \mathcal{D} :



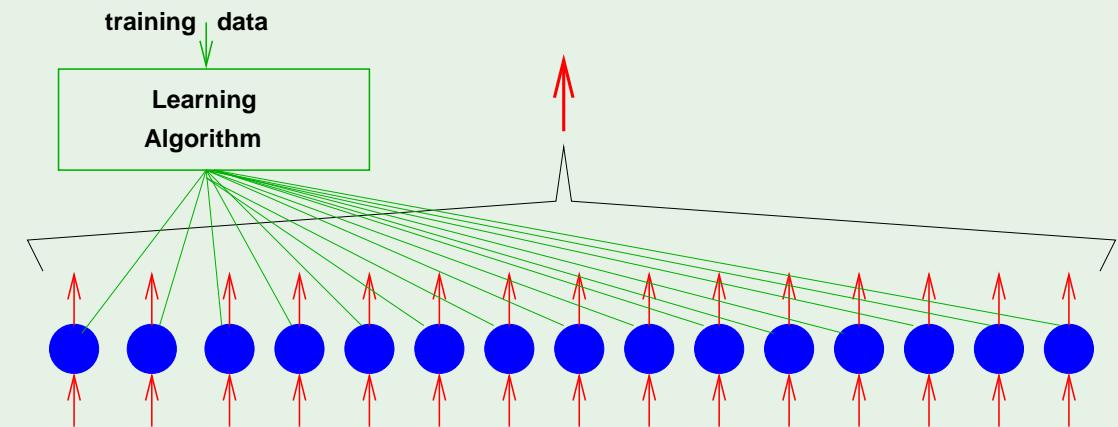
Regression: take an average

Classification: take a vote

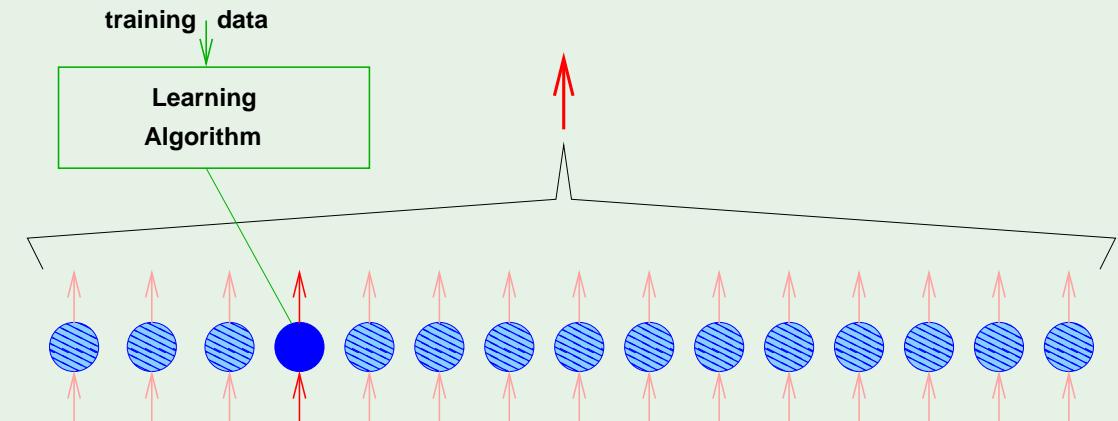
a.k.a. *ensemble learning* and *boosting*

Different from 2-layer learning

In a 2-layer model, all units learn **jointly**:



In aggregation, they learn **independently** then get combined:



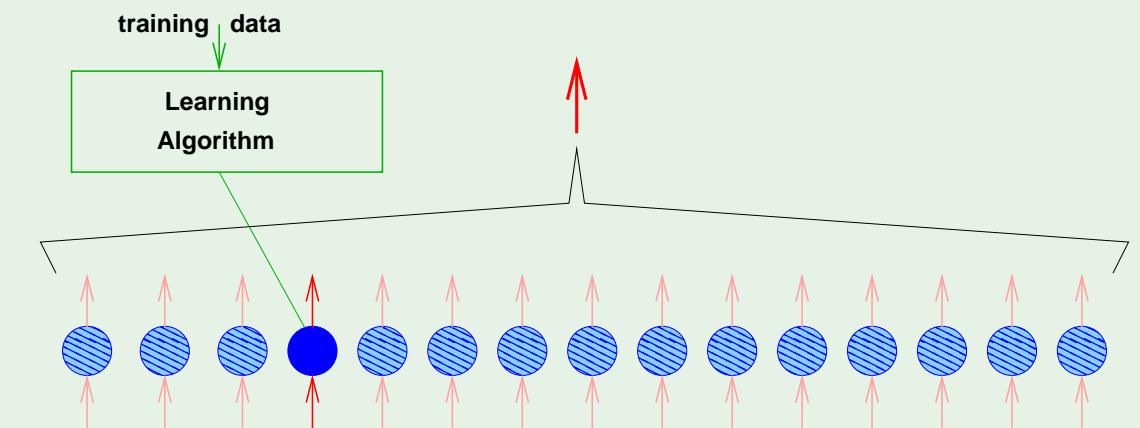
Two types of aggregation

1. **After the fact:** combines existing solutions

Example. Netflix teams merging “blending”

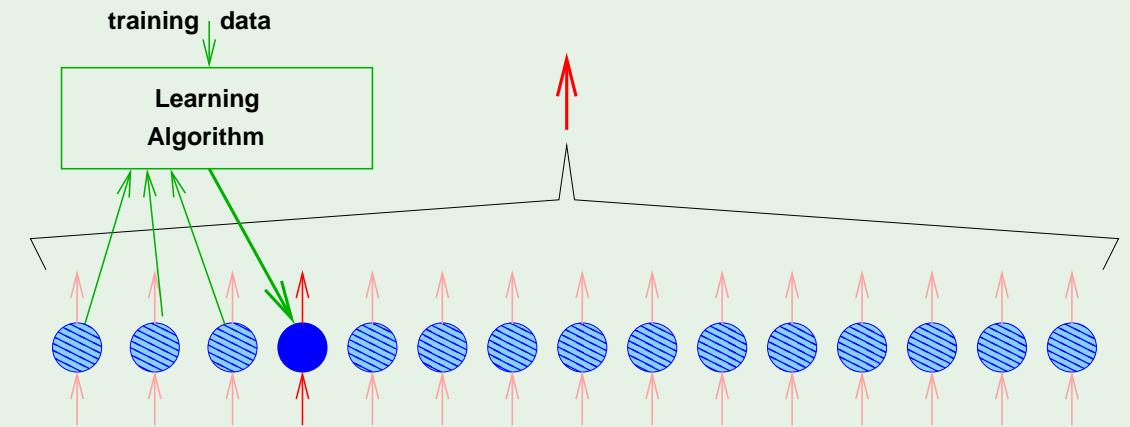
2. **Before the fact:** creates solutions to be combined

Example. Bagging - resampling \mathcal{D}



Decorrelation - boosting

Create h_1, \dots, h_t, \dots sequentially: Make h_t decorrelated with previous h 's:



Emphasize points in \mathcal{D} that were misclassified

Choose weight of h_t based on $E_{\text{in}}(h_t)$

Blending - after the fact

For regression, $h_1, h_2, \dots, h_T \longrightarrow g(\mathbf{x}) = \sum_{t=1}^T \alpha_t h_t(\mathbf{x})$

Principled choice of α_t 's: minimize the error on an “aggregation data set” pseudo-inverse

Some α_t 's can come out negative

Most valuable h_t in the blend?

Outline

- The map of machine learning
- Bayesian learning
- Aggregation methods
- Acknowledgments

Course content

Professor **Malik Magdon-Ismail**, *RPI*

Professor **Hsuan-Tien Lin**, *NTU*

Course staff

Carlos Gonzalez (*Head TA*)

Ron Appel

Costis Sideris

Doris Xin

Filming, production, and infrastructure

Leslie Maxfield and the **AMT** staff

Rich Fagen and the **IMSS** staff

Caltech support

IST - Mathieu Desbrun

E&AS Division - Ares Rosakis and Mani Chandy

Provost's Office - Ed Stolper and Melany Hunt

Many others

Caltech TA's and staff members

Caltech alumni and Alumni Association

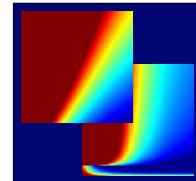
Colleagues all over the world

To the fond memory of

Faiza A. Ibrahim

Learning From Data*Caltech*<http://work.caltech.edu/telecourse.html>

2012



Online Homework # 1

Collaboration in the sense of discussions is allowed, but you should NOT discuss your selected answers with anyone. Books and notes can be consulted. All questions will have multiple choice answers, the choices being labeled [a], [b], [c], [d], [e]. Most questions will have 5 possible choices, although some may have fewer choices. You should enter your solutions online by logging into your account at the course web site.

Note about the homeworks

- The goal of the homeworks is to facilitate a deeper understanding of the course material. The questions are not designed to be puzzles with catchy answers. They are meant to make you roll your sleeves, face uncertainties, and approach the problem from different angles.
- The problems range from easy to hard, and from theoretical to practical. Some problems require running a full experiment to arrive at the answer.
- The answer may not be obvious or numerically close to one of the choices. The intent is to prompt discussion and exchange of ideas.
- Speaking of discussion, you are encouraged to take part in the forum

<http://book.caltech.edu/bookforum>

where there are many threads about each homework. We hope that you will contribute to the discussion as well.

- Please follow the forum guidelines for posting answers (see the “BEFORE posting answers” top thread there).

The Learning Problem

1. What types of learning, if any, best describe the following three scenarios:
 - (i) A coin classification system is created for a vending machine. In order to do this, the developers obtain exact coin specifications from the U.S. Mint and derive a statistical model of the size, weight, and denomination, which the vending machine then uses to classify its coins.
 - (ii) Instead of calling the U.S. Mint to obtain coin information, an algorithm is presented with a large set of labeled coins. The algorithm uses this data to infer decision boundaries which the vending machine then uses to classify its coins.
 - (iii) A computer develops a strategy for playing Tic-Tac-Toe by playing repeatedly and adjusting its strategy by penalizing moves that eventually lead to losing.
 - [a] (i) Supervised Learning, (ii) Unsupervised Learning, (iii) Reinforcement Learning
 - [b] (i) Supervised Learning, (ii) Not learning, (iii) Unsupervised Learning
 - [c] (i) Not learning, (ii) Reinforcement Learning, (iii) Supervised Learning
 - [d] (i) Not learning, (ii) Supervised Learning, (iii) Reinforcement Learning
 - [e] (i) Supervised Learning, (ii) Reinforcement Learning, (iii) Unsupervised Learning
2. Which of the following problems are best suited for the learning approach?
 - (i) Classifying numbers into primes and non-primes.
 - (ii) Detecting potential fraud in credit card charges.
 - (iii) Determining the time it would take a falling object to hit the ground.
 - (iv) Determining the optimal cycle for traffic lights in a busy intersection.
 - [a] (ii) and (iv)
 - [b] (i) and (ii)
 - [c] (i), (ii), and (iii).
 - [d] (iii)

- [e] (i) and (iii)

Bins and Marbles

3. We have 2 opaque bags, each containing 2 balls. One bag has 2 black balls and the other has a black ball and a white ball. You pick a bag at random and then pick one of the balls in that bag at random. When you look at the ball, it is black. You now pick the second ball from that same bag. What is the probability that this ball is also black?

[a] $1/4$

[b] $1/3$

[c] $1/2$

[d] $2/3$

[e] $3/4$

Consider a sample of 10 marbles drawn from a bin that has red and green marbles. The probability that any marble we draw is red is $\mu = 0.55$ (independently, with replacement). We address the probability of getting no red marbles ($\nu = 0$) in the following cases:

4. We draw only one such sample. Compute the probability that $\nu = 0$. The closest answer is (closest is the answer that makes the expression $|$ your answer – given option| closest to 0):

[a] 7.331×10^{-6}

[b] 3.405×10^{-4}

[c] 0.289

[d] 0.450

[e] 0.550

5. We draw 1,000 independent samples. Compute the probability that (at least) one of the samples has $\nu = 0$. The closest answer is:

[a] 7.331×10^{-6}

[b] 3.405×10^{-4}

[c] 0.289

[d] 0.450

[e] 0.550

Feasibility of Learning

Consider a boolean target function over a 3-dimensional input space $\mathcal{X} = \{0, 1\}^3$ (instead of our ± 1 binary convention, we use 0,1 here since it is standard for boolean functions). We are given a data set \mathcal{D} of five examples represented in the table below, where $y_n = f(\mathbf{x}_n)$ for $n = 1, 2, 3, 4, 5$.

\mathbf{x}_n			y_n
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1

Note that in this simple boolean case, we can enumerate the entire input space (since there are only $2^3 = 8$ distinct input vectors), and we can enumerate the set of all possible target functions (there are only $2^{2^3} = 256$ distinct boolean function on 3 boolean inputs).

Let us look at the problem of learning f . Since f is unknown except inside \mathcal{D} , any function that agrees with \mathcal{D} could conceivably be f . Since there are only 3 points in \mathcal{X} outside \mathcal{D} , there are only $2^3 = 8$ such functions.

The remaining points in \mathcal{X} which are not in \mathcal{D} are: 101, 110, and 111. We want to determine the hypothesis that agrees the most with the possible target functions. In order to quantify this, count how many of the 8 possible target functions agree with each hypothesis on all 3 points, how many agree with just 2 of the points, with just 1, and how many do not agree on any points. The final score for each hypothesis is computed as follows:

Score = (# of target functions agreeing with hypothesis on all 3 points) * 3 + (# of target functions agreeing with hypothesis on 2 points) * 2 + (# of target functions agreeing with hypothesis on 1 points)*1 + (# of target functions agreeing with hypothesis on 0 points)*0.

6. Which hypothesis g agrees the most with the possible target functions in terms of the above score?

[a] g returns 1 for all three points.

[b] g returns 0 for all three points.

- [c] g is the XOR function applied to \mathbf{x} , i.e., if the number of 1's in \mathbf{x} is odd, g returns 1; if it is even, g returns 0.
- [d] g returns the opposite of the XOR function: if number of 1's is odd, it returns 0, otherwise returns 1.
- [e] They are all equivalent.

The Perceptron Learning Algorithm

In this problem you will create your own target function f and data set \mathcal{D} to see how the Perceptron Learning Algorithm works. Take $d = 2$ so you can visualize the problem, and choose a random line in the plane as your target function f (do this by taking two random, uniformly distributed points on $[-1, 1] \times [-1, 1]$ and taking the line passing through them), where one side of the line maps to +1 and the other maps to -1. Choose the inputs \mathbf{x}_n of the data set as random points in the plane, and evaluate the target function on each \mathbf{x}_n to get the corresponding output y_n .

7. Take $\mathcal{X} = [-1, 1] \times [-1, 1]$ with uniform probability of picking each $\mathbf{x} \in \mathcal{X}$. Take $N = 10$. Run the Perceptron Learning Algorithm to find g and measure the difference between f and g as $\Pr(f(\mathbf{x}) \neq g(\mathbf{x}))$ (you can either calculate this exactly, or approximate it by generating a sufficiently large separate set of points to evaluate it). Repeat the experiment 1000 times and take the average. Start the PLA with the weight vector \mathbf{w} being all zeros, and at each iteration have the algorithm choose a point randomly from the set of misclassified points.

How many iterations does it take on average for the PLA to converge for $N = 10$ training points? Pick the value closest to your results (again, closest is the answer that makes the expression $|your\ answer - given\ option|$ closest to 0).

- [a] 1
 - [b] 15
 - [c] 300
 - [d] 5000
 - [e] 10000
8. Which of the following is closest to $\Pr(f(\mathbf{x}) \neq g(\mathbf{x}))$ for $N = 10$?
- [a] 0.001
 - [b] 0.01
 - [c] 0.1

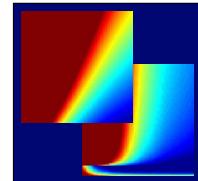
- [d] 0.5
 - [e] 0.8
9. Now, try $N = 100$. How many iterations does it take on average for the PLA to converge for $N=100$ training points? Pick the value closest to your results.
- [a] 50
 - [b] 100
 - [c] 500
 - [d] 1000
 - [e] 5000
10. Which of the following is closest to $\Pr(f(\mathbf{x}) \neq g(\mathbf{x}))$ for $N = 100$?
- [a] 0.001
 - [b] 0.01
 - [c] 0.1
 - [d] 0.5
 - [e] 0.8

Learning From Data

Caltech

<http://work.caltech.edu/telecourse.html>

2012



Answer Key To Online Homework # 1

1. [d]
2. [a]
3. [d]
4. [b]
5. [c]
6. [e]
7. [b]
8. [c]
9. [b]
10. [b]

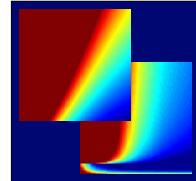
Visit the forum (<http://book.caltech.edu/bookforum>) for discussion. Please follow the forum guidelines for posting answers (see the “BEFORE posting answers” top thread there).

Learning From Data

Yaser Abu-Mostafa, *Caltech*

<http://work.caltech.edu/telecourse>

Self-paced version



Homework # 2

All questions have multiple-choice answers ([a], [b], [c], ...). You can collaborate with others, but do not discuss the selected or excluded choices in the answers. You can consult books and notes, but not other people's solutions. Your solutions should be based on your own work. Definitions and notation follow the lectures.

Note about the homework

- The goal of the homework is to facilitate a deeper understanding of the course material. The questions are not designed to be puzzles with catchy answers. They are meant to make you roll up your sleeves, face uncertainties, and approach the problem from different angles.
- The problems range from easy to difficult, and from practical to theoretical. Some problems require running a full experiment to arrive at the answer.
- The answer may not be obvious or numerically close to one of the choices, but one (and only one) choice will be correct if you follow the instructions precisely in each problem. You are encouraged to explore the problem further by experimenting with variations on these instructions, for the learning benefit.
- You are also encouraged to take part in the forum

<http://book.caltech.edu/bookforum>

where there are many threads about each homework set. We hope that you will contribute to the discussion as well. Please follow the forum guidelines for posting answers (see the “BEFORE posting answers” announcement at the top there).

© 2012-2015 Yaser Abu-Mostafa. All rights reserved. No redistribution in any format. No translation or derivative products without written permission.

● Hoeffding Inequality

Run a computer simulation for flipping 1,000 virtual fair coins. Flip each coin independently 10 times. Focus on 3 coins as follows: c_1 is the first coin flipped, c_{rand} is a coin chosen randomly from the 1,000, and c_{\min} is the coin which had the minimum frequency of heads (pick the earlier one in case of a tie). Let ν_1 , ν_{rand} , and ν_{\min} be the *fraction* of heads obtained for the 3 respective coins out of the 10 tosses.

Run the experiment 100,000 times in order to get a full distribution of ν_1 , ν_{rand} , and ν_{\min} (note that c_{rand} and c_{\min} will change from run to run).

1. The average value of ν_{\min} is closest to:

- [a] 0
- [b] 0.01
- [c] 0.1
- [d] 0.5
- [e] 0.67

2. Which coin(s) has a distribution of ν that satisfies the (single-bin) Hoeffding Inequality?

- [a] c_1 only
- [b] c_{rand} only
- [c] c_{\min} only
- [d] c_1 and c_{rand}
- [e] c_{\min} and c_{rand}

● Error and Noise

Consider the bin model for a hypothesis h that makes an error with probability μ in approximating a deterministic target function f (both h and f are binary functions). If we use the same h to approximate a noisy version of f given by:

$$P(y \mid \mathbf{x}) = \begin{cases} \lambda & y = f(x) \\ 1 - \lambda & y \neq f(x) \end{cases}$$

3. What is the probability of error that h makes in approximating y ? *Hint: Two wrongs can make a right!*

- [a] μ
 - [b] λ
 - [c] $1-\mu$
 - [d] $(1-\lambda)*\mu + \lambda*(1-\mu)$
 - [e] $(1-\lambda)*(1-\mu) + \lambda*\mu$
4. At what value of λ will the performance of h be independent of μ ?

- [a] 0
- [b] 0.5
- [c] $1/\sqrt{2}$
- [d] 1
- [e] No values of λ

● Linear Regression

In these problems, we will explore how Linear Regression for classification works. As with the Perceptron Learning Algorithm in Homework # 1, you will create your own target function f and data set \mathcal{D} . Take $d = 2$ so you can visualize the problem, and assume $\mathcal{X} = [-1, 1] \times [-1, 1]$ with uniform probability of picking each $\mathbf{x} \in \mathcal{X}$. In each run, choose a random line in the plane as your target function f (do this by taking two random, uniformly distributed points in $[-1, 1] \times [-1, 1]$ and taking the line passing through them), where one side of the line maps to $+1$ and the other maps to -1 . Choose the inputs \mathbf{x}_n of the data set as random points (uniformly in \mathcal{X}), and evaluate the target function on each \mathbf{x}_n to get the corresponding output y_n .

5. Take $N = 100$. Use Linear Regression to find g and evaluate E_{in} , the fraction of in-sample points which got classified incorrectly. Repeat the experiment 1000 times and take the average (keep the g 's as they will be used again in Problem 6). Which of the following values is closest to the average E_{in} ? (*Closest* is the option that makes the expression $|\text{your answer} - \text{given option}|$ closest to 0. Use this definition of *closest* here and throughout.)

- [a] 0
- [b] 0.001
- [c] 0.01
- [d] 0.1
- [e] 0.5

6. Now, generate 1000 fresh points and use them to estimate the out-of-sample error E_{out} of g that you got in Problem 5 (number of misclassified out-of-sample points / total number of out-of-sample points). Again, run the experiment 1000 times and take the average. Which value is closest to the average E_{out} ?

- [a] 0
- [b] 0.001
- [c] 0.01
- [d] 0.1
- [e] 0.5

7. Now, take $N = 10$. After finding the weights using Linear Regression, use them as a vector of initial weights for the Perceptron Learning Algorithm. Run PLA until it converges to a final vector of weights that completely separates all the in-sample points. Among the choices below, what is the closest value to the average number of iterations (over 1000 runs) that PLA takes to converge? (When implementing PLA, have the algorithm choose a point randomly from the set of misclassified points at each iteration)

- [a] 1
- [b] 15
- [c] 300
- [d] 5000
- [e] 10000

● Nonlinear Transformation

In these problems, we again apply Linear Regression for classification. Consider the target function:

$$f(x_1, x_2) = \text{sign}(x_1^2 + x_2^2 - 0.6)$$

Generate a training set of $N = 1000$ points on $\mathcal{X} = [-1, 1] \times [-1, 1]$ with a uniform probability of picking each $\mathbf{x} \in \mathcal{X}$. Generate simulated noise by flipping the sign of the output in a randomly selected 10% subset of the generated training set.

8. Carry out Linear Regression without transformation, i.e., with feature vector:

$$(1, x_1, x_2),$$

to find the weight \mathbf{w} . What is the closest value to the classification in-sample error E_{in} ? (Run the experiment 1000 times and take the average E_{in} to reduce variation in your results.)

- [a] 0
- [b] 0.1
- [c] 0.3
- [d] 0.5
- [e] 0.8

9. Now, transform the $N = 1000$ training data into the following nonlinear feature vector:

$$(1, x_1, x_2, x_1x_2, x_1^2, x_2^2)$$

Find the vector $\tilde{\mathbf{w}}$ that corresponds to the solution of Linear Regression. Which of the following hypotheses is closest to the one you find? Closest here means agrees the most with your hypothesis (has the highest probability of agreeing on a randomly selected point). Average over a few runs to make sure your answer is stable.

- [a] $g(x_1, x_2) = \text{sign}(-1 - 0.05x_1 + 0.08x_2 + 0.13x_1x_2 + 1.5x_1^2 + 1.5x_2^2)$
- [b] $g(x_1, x_2) = \text{sign}(-1 - 0.05x_1 + 0.08x_2 + 0.13x_1x_2 + 1.5x_1^2 + 15x_2^2)$
- [c] $g(x_1, x_2) = \text{sign}(-1 - 0.05x_1 + 0.08x_2 + 0.13x_1x_2 + 15x_1^2 + 1.5x_2^2)$
- [d] $g(x_1, x_2) = \text{sign}(-1 - 1.5x_1 + 0.08x_2 + 0.13x_1x_2 + 0.05x_1^2 + 0.05x_2^2)$
- [e] $g(x_1, x_2) = \text{sign}(-1 - 0.05x_1 + 0.08x_2 + 1.5x_1x_2 + 0.15x_1^2 + 0.15x_2^2)$

10. What is the closest value to the classification out-of-sample error E_{out} of your hypothesis from Problem 9? (Estimate it by generating a new set of 1000 points and adding noise, as before. Average over 1000 runs to reduce the variation in your results.)

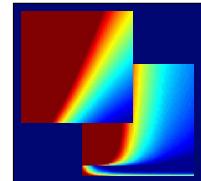
- [a] 0
- [b] 0.1
- [c] 0.3
- [d] 0.5
- [e] 0.8

Learning From Data

Caltech

<http://work.caltech.edu/telecourse.html>

2012



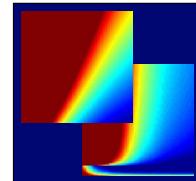
Answer Key To Online Homework # 2

1. [b]
2. [d]
3. [e]
4. [b]
5. [c]
6. [c]
7. [a]
8. [d]
9. [a]
10. [b]

Visit the forum (<http://book.caltech.edu/bookforum>) for discussion. Please follow the forum guidelines for posting answers (see the “BEFORE posting answers” top thread there).

Learning From Data*Caltech*<http://work.caltech.edu/telecourse.html>

2012



Online Homework # 3

Collaboration in the sense of discussions is allowed, but you should NOT discuss your selected answers with anyone. Books and notes can be consulted. All questions will have multiple choice answers, the choices being labeled [a], [b], [c], [d], [e]. Most questions will have 5 possible choices, although some may have fewer choices. You should enter your solutions online by logging into your account at the course web site.

Note about the homeworks

- The goal of the homeworks is to facilitate a deeper understanding of the course material. The questions are not designed to be puzzles with catchy answers. They are meant to make you roll your sleeves, face uncertainties, and approach the problem from different angles.
- The problems range from easy to hard, and from theoretical to practical. Some problems require running a full experiment to arrive at the answer.
- The answer may not be obvious or numerically close to one of the choices. The intent is to prompt discussion and exchange of ideas.
- Speaking of discussion, you are encouraged to take part in the forum

<http://book.caltech.edu/bookforum>

where there are many threads about each homework. We hope that you will contribute to the discussion as well.

- Please follow the forum guidelines for posting answers (see the “BEFORE posting answers” top thread there).

Generalization Error

1. The modified Hoeffding Inequality provides a way to characterize the generalization error with a probabilistic bound

$$\mathbb{P}[|E_{in}(g) - E_{out}(g)| > \epsilon] \leq 2Me^{-2\epsilon^2 N}$$

for any $\epsilon > 0$. If we set $\epsilon = 0.05$ and want the probability bound $2Me^{-2\epsilon^2 N}$ to be at most 0.03, what is the least number of examples N (among the choices given) needed for the case $M = 1$?

- [a] 500
- [b] 1000
- [c] 1500
- [d] 2000
- [e] More examples are needed

2. Repeat for the case $M = 10$.

- [a] 500
- [b] 1000
- [c] 1500
- [d] 2000
- [e] More examples are needed

3. Repeat for the case $M = 100$.

- [a] 500
- [b] 1000
- [c] 1500
- [d] 2000
- [e] More examples are needed

Break Point

4. As shown in class, the (smallest) break point for the Perceptron Model in the two-dimensional case (\mathbb{R}^2) is 4 points. What is the smallest break point for the Perceptron Model in \mathbb{R}^3 ? (i.e., instead of hypothesis consisting of separating lines, they are separating planes.)

[a] 4

[b] 5

[c] 6

[d] 7

[e] 8

Growth Function

5. Which of the following are possible formulas for a growth function $m_{\mathcal{H}}(N)$:

- i) $1 + N$
- ii) $1 + N + \binom{N}{2}$
- iii) $\sum_{i=1}^{\lfloor \sqrt{N} \rfloor} \binom{N}{i}$
- iv) $2^{\lfloor N/2 \rfloor}$
- v) 2^N

[a] i, v

[b] i, ii, v

[c] i, iv, v

[d] i, ii, iii, v

[e] i, ii, iii, iv, v

Fun with Intervals

6. Consider the “2-intervals” learning model, where $h: \mathbb{R} \rightarrow \{-1, +1\}$ and $h(x) = +1$ if the point is within either of two arbitrarily chosen intervals and -1 otherwise. What is the (smallest) break point of this hypothesis set?

[a] 3

[b] 4

[c] 5

[d] 6

[e] 7

7. Which of the following is the growth function $m_H(N)$ for the “2-intervals” hypothesis set?

[a] $\binom{N+1}{4}$

[b] $\binom{N+1}{2} + 1$

[c] $\binom{N+1}{4} + \binom{N+1}{2} + 1$

[d] $\binom{N+1}{4} + \binom{N+1}{3} + \binom{N+1}{2} + \binom{N+1}{1} + 1$

[e] None of the above

8. Now, consider the general case: the “M-intervals” learning model. Again $h : \mathbb{R} \rightarrow \{-1, +1\}$, where $h(x) = +1$ if the point falls inside any of M arbitrarily chosen intervals, otherwise $h(x) = -1$. What is the (smallest) break point of this hypothesis set?

[a] M

[b] $M + 1$

[c] M^2

[d] $2M + 1$

[e] $2M - 1$

Convex Sets: The Triangle

9. Consider the “triangle” learning model, where $h : \mathbb{R}^2 \rightarrow \{-1, +1\}$ and $h(x) = +1$ if x lies within an arbitrarily chosen triangle in the plane and -1 otherwise. Which among the following choices is the largest number of points in \mathbb{R}^2 that can be shattered by this hypothesis set?

[a] 1

[b] 3

[c] 5

[d] 7

[e] 9

Non-Convex Sets: Concentric Circles

10. Compute the growth function $m_{\mathcal{H}}(N)$ for the learning model made up of two concentric circles in \mathbb{R}^2 : \mathcal{H} contains the functions which are +1 for

$$a^2 \leq x_1^2 + x_2^2 \leq b^2$$

and -1 otherwise. The growth function is

[a] $N + 1$

[b] $\binom{N+1}{2} + 1$

[c] $\binom{N+1}{3} + 1$

[d] $2N^2 + 1$

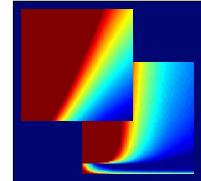
[e] None of the above

Learning From Data

Caltech

<http://work.caltech.edu/telecourse.html>

2012



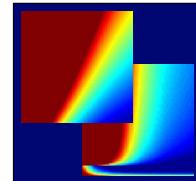
Answer Key To Online Homework # 3

1. [b]
2. [c]
3. [d]
4. [b]
5. [b]
6. [c]
7. [c]
8. [d]
9. [d]
10. [b]

Visit the forum (<http://book.caltech.edu/bookforum>) for discussion. Please follow the forum guidelines for posting answers (see the “BEFORE posting answers” top thread there).

Learning From Data*Caltech*<http://work.caltech.edu/telecourse.html>

2012



Online Homework # 4

Collaboration in the sense of discussions is allowed, but you should NOT discuss your selected answers with anyone. Books and notes can be consulted. All questions will have multiple choice answers, the choices being labeled [a], [b], [c], [d], [e]. Most questions will have 5 possible choices, although some may have fewer choices. You should enter your solutions online by logging into your account at the course web site.

Note about the homeworks

- The goal of the homeworks is to facilitate a deeper understanding of the course material. The questions are not designed to be puzzles with catchy answers. They are meant to make you roll your sleeves, face uncertainties, and approach the problem from different angles.
- The problems range from easy to hard, and from theoretical to practical. Some problems require running a full experiment to arrive at the answer.
- The answer may not be obvious or numerically close to one of the choices. The intent is to prompt discussion and exchange of ideas.
- Speaking of discussion, you are encouraged to take part in the forum

<http://book.caltech.edu/bookforum>

where there are many threads about each homework. We hope that you will contribute to the discussion as well.

- Please follow the forum guidelines for posting answers (see the “BEFORE posting answers” top thread there).

Generalization Error

In Problems 1-3, we look at generalization bounds numerically. For $N > d_{\text{VC}}$, use the simple approximate bound $N^{d_{\text{VC}}}$ for the growth function.

1. For an \mathcal{H} with $d_{\text{VC}} = 10$, what is the closest numerical approximation of the sample size that the VC generalization bound predicts if you want 95% confidence that your generalization error is at most 0.05?

[a] 420,000

[b] 430,000

[c] 440,000

[d] 450,000

[e] 460,000

2. There are a number of bounds on the generalization error ϵ , all holding with probability at least $1 - \delta$. Fix $d_{\text{VC}} = 50$ and $\delta = 0.05$ and plot these bounds as a function of N . Which bound is the smallest for very large N ? Note that [c] and [d] are implicit bounds in ϵ .

[a] Original VC bound: $\epsilon \leq \sqrt{\frac{8}{N} \ln \frac{4m_{\mathcal{H}}(2N)}{\delta}}$

[b] Rademacher Penalty Bound: $\epsilon \leq \sqrt{\frac{2 \ln(2Nm_{\mathcal{H}}(N))}{N}} + \sqrt{\frac{2}{N} \ln \frac{1}{\delta}} + \frac{1}{N}$

[c] Parrondo and Van den Broek: $\epsilon \leq \sqrt{\frac{1}{N}(2\epsilon + \ln \frac{6m_{\mathcal{H}}(2N)}{\delta})}$

[d] Devroye: $\epsilon \leq \sqrt{\frac{1}{2N}(4\epsilon(1 + \epsilon) + \ln \frac{4m_{\mathcal{H}}(N^2)}{\delta})}$

[e] They are all equal.

3. For small N , say $N = 5$, which bound is the smallest?

[a] Original VC bound: $\epsilon \leq \sqrt{\frac{8}{N} \ln \frac{4m_{\mathcal{H}}(2N)}{\delta}}$

[b] Rademacher Penalty Bound: $\epsilon \leq \sqrt{\frac{2 \ln(2Nm_{\mathcal{H}}(N))}{N}} + \sqrt{\frac{2}{N} \ln \frac{1}{\delta}} + \frac{1}{N}$

[c] Parrondo and Van den Broek: $\epsilon \leq \sqrt{\frac{1}{N}(2\epsilon + \ln \frac{6m_{\mathcal{H}}(2N)}{\delta})}$

[d] Devroye: $\epsilon \leq \sqrt{\frac{1}{2N}(4\epsilon(1 + \epsilon) + \ln \frac{4m_{\mathcal{H}}(N^2)}{\delta})}$

- [e] They are all equal.

Bias and Variance

Consider the case where the target function $f : [-1, 1] \rightarrow \mathbb{R}$ is given by $f(x) = \sin(\pi x)$ and the input probability distribution is uniform on $[-1, 1]$. Assume that the training set has only two examples (picked independently), and that the learning algorithm picks the hypothesis that minimizes the mean squared error on the examples.

4. Assume the learning model consists of all hypotheses of the form $h(x) = ax$. What is the expected value of the hypothesis, $\bar{g}(x)$, that the learning algorithm produces (expected value with respect to the data set)? Coefficients are given to second decimal digit accuracy.

- [a] $\bar{g}(x) = 0$
- [b] $\bar{g}(x) = 0.13x$
- [c] $\bar{g}(x) = 0.45x$
- [d] $\bar{g}(x) = 0.79x$
- [e] None of the above

5. What is the closest value to the bias in this case?

- [a] 0.1
- [b] 0.3
- [c] 0.5
- [d] 0.7
- [e] 1.0

6. What is the closest value to the variance in this case?

- [a] 0.2
- [b] 0.4
- [c] 0.6
- [d] 0.8

[e] 1.0

7. Now, let's change \mathcal{H} . Which of the following learning models has the least expected value of out-of-sample error?

- [a] Hypotheses of the form $h(x) = b$.
- [b] Hypotheses of the form $h(x) = ax$.
- [c] Hypotheses of the form $h(x) = ax + b$.
- [d] Hypotheses of the form $h(x) = ax^2$.
- [e] Hypotheses of the form $h(x) = ax^2 + b$.

VC Dimension

8. Assume $q \geq 1$ is an integer and let $m_{\mathcal{H}}(1) = 2$. What is the VC dimension of a hypothesis set whose growth function satisfies: $m_{\mathcal{H}}(N+1) = 2m_{\mathcal{H}}(N) - \binom{N}{q}$ (*Hint: there is no way to choose 10 people from a group of 7 people*)

- [a] $q - 2$
- [b] $q - 1$
- [c] q
- [d] $q + 1$
- [e] None of the above

9. For hypothesis sets $\mathcal{H}_1, \mathcal{H}_2, \dots, \mathcal{H}_K$ with finite VC dimensions $d_{\text{VC}}(\mathcal{H}_k)$, some of the following bounds are correct and some are not. Which among the correct ones is the tightest bound (the smallest range of values) on the VC dimension of the **intersection** of the sets: $d_{\text{VC}}(\bigcap_{k=1}^K \mathcal{H}_k)$?

- [a] $0 \leq d_{\text{VC}}(\bigcap_{k=1}^K \mathcal{H}_k) \leq \sum_{k=1}^K d_{\text{VC}}(\mathcal{H}_k)$
- [b] $0 \leq d_{\text{VC}}(\bigcap_{k=1}^K \mathcal{H}_k) \leq \min\{d_{\text{VC}}(\mathcal{H}_k)\}_{k=1}^K$
- [c] $0 \leq d_{\text{VC}}(\bigcap_{k=1}^K \mathcal{H}_k) \leq \max\{d_{\text{VC}}(\mathcal{H}_k)\}_{k=1}^K$
- [d] $\min\{d_{\text{VC}}(\mathcal{H}_k)\}_{k=1}^K \leq d_{\text{VC}}(\bigcap_{k=1}^K \mathcal{H}_k) \leq \max\{d_{\text{VC}}(\mathcal{H}_k)\}_{k=1}^K$

$$[\text{e}] \quad \min\{d_{\text{VC}}(\mathcal{H}_k)\}_{k=1}^K \leq d_{\text{VC}}(\bigcap_{k=1}^K \mathcal{H}_k) \leq \sum_{k=1}^K d_{\text{VC}}(\mathcal{H}_k)$$

10. For hypothesis sets $\mathcal{H}_1, \mathcal{H}_2, \dots, \mathcal{H}_K$ with finite VC dimensions $d_{\text{VC}}(\mathcal{H}_k)$, some of the following bounds are correct and some are not. Which among the correct ones is the tightest bound (the smallest range of values) on the VC dimension of the **union** of the sets: $d_{\text{VC}}(\bigcup_{k=1}^K \mathcal{H}_k)$?

$$[\text{a}] \quad 0 \leq d_{\text{VC}}(\bigcup_{k=1}^K \mathcal{H}_k) \leq \sum_{k=1}^K d_{\text{VC}}(\mathcal{H}_k)$$

$$[\text{b}] \quad 0 \leq d_{\text{VC}}(\bigcup_{k=1}^K \mathcal{H}_k) \leq K - 1 + \sum_{k=1}^K d_{\text{VC}}(\mathcal{H}_k)$$

$$[\text{c}] \quad \min\{d_{\text{VC}}(\mathcal{H}_k)\}_{k=1}^K \leq d_{\text{VC}}(\bigcup_{k=1}^K \mathcal{H}_k) \leq \sum_{k=1}^K d_{\text{VC}}(\mathcal{H}_k)$$

$$[\text{d}] \quad \max\{d_{\text{VC}}(\mathcal{H}_k)\}_{k=1}^K \leq d_{\text{VC}}(\bigcup_{k=1}^K \mathcal{H}_k) \leq \sum_{k=1}^K d_{\text{VC}}(\mathcal{H}_k)$$

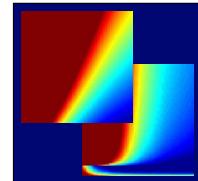
$$[\text{e}] \quad \max\{d_{\text{VC}}(\mathcal{H}_k)\}_{k=1}^K \leq d_{\text{VC}}(\bigcup_{k=1}^K \mathcal{H}_k) \leq K - 1 + \sum_{k=1}^K d_{\text{VC}}(\mathcal{H}_k)$$

Learning From Data

Caltech

<http://work.caltech.edu/telecourse.html>

2012



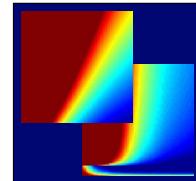
Answer Key To Online Homework # 4

1. [d]
2. [d]
3. [c]
4. [e]
5. [b]
6. [a]
7. [b]
8. [c]
9. [b]
10. [e]

Visit the forum (<http://book.caltech.edu/bookforum>) for discussion. Please follow the forum guidelines for posting answers (see the “BEFORE posting answers” top thread there).

Learning From Data*Caltech*<http://work.caltech.edu/telecourse.html>

2012



Online Homework # 5

Collaboration in the sense of discussions is allowed, but you should NOT discuss your selected answers with anyone. Books and notes can be consulted. All questions will have multiple choice answers, the choices being labeled [a], [b], [c], [d], [e]. Most questions will have 5 possible choices, although some may have fewer choices. You should enter your solutions online by logging into your account at the course web site.

Note about the homeworks

- The goal of the homeworks is to facilitate a deeper understanding of the course material. The questions are not designed to be puzzles with catchy answers. They are meant to make you roll your sleeves, face uncertainties, and approach the problem from different angles.
- The problems range from easy to hard, and from theoretical to practical. Some problems require running a full experiment to arrive at the answer.
- The answer may not be obvious or numerically close to one of the choices. The intent is to prompt discussion and exchange of ideas.
- Speaking of discussion, you are encouraged to take part in the forum

<http://book.caltech.edu/bookforum>

where there are many threads about each homework. We hope that you will contribute to the discussion as well.

- Please follow the forum guidelines for posting answers (see the “BEFORE posting answers” top thread there).

Linear Regression Error

Consider a noisy target $y = \mathbf{w}^*{}^T \mathbf{x} + \epsilon$, where $\mathbf{x} \in \mathbb{R}^d$ (with the added coordinate $x_0 = 1$), $y \in \mathbb{R}$, \mathbf{w}^* is an unknown vector, and ϵ is a noise term with zero mean and σ^2 variance. Assume ϵ is independent of \mathbf{x} and of all other ϵ 's. If linear regression is carried out using a training data set $\mathcal{D} = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)\}$, and outputs the parameter vector \mathbf{w}_{lin} , it can be shown that the expected in-sample error E_{in} with respect to \mathcal{D} is given by:

$$\mathbb{E}_{\mathcal{D}}[E_{\text{in}}(\mathbf{w}_{\text{lin}})] = \sigma^2 \left(1 - \frac{d+1}{N}\right)$$

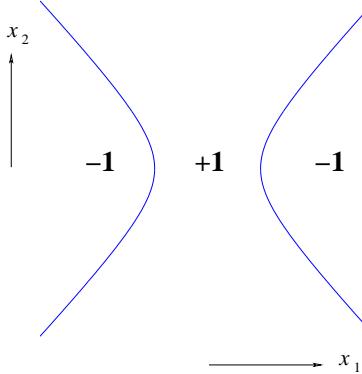
1. For $\sigma = 0.1$ and $d = 8$, which among the following choices is the smallest number of examples N that will result in an expected E_{in} greater than 0.008?
 - [a] 10
 - [b] 25
 - [c] 100
 - [d] 500
 - [e] 1000

Nonlinear Transforms

Consider the feature transform $\Phi : \mathbb{R}^2 \rightarrow \mathbb{R}^2$ (plus the added zeroth coordinate) given by:

$$\Phi(1, x_1, x_2) = (1, x_1^2, x_2^2)$$

2. Which of the following sets of constraints on the weights in the \mathcal{Z} space could correspond to the hyperbolic decision boundary in \mathcal{X} depicted in the figure? You may assume that \tilde{w}_0 can be selected to achieve the desired boundary.



- [a] $\tilde{w}_1 = 0, \tilde{w}_2 > 0$
- [b] $\tilde{w}_1 > 0, \tilde{w}_2 = 0$
- [c] $\tilde{w}_1 > 0, \tilde{w}_2 > 0$
- [d] $\tilde{w}_1 < 0, \tilde{w}_2 > 0$
- [e] $\tilde{w}_1 > 0, \tilde{w}_2 < 0$

Consider the 4th order polynomial transform from the input space \mathbb{R}^2 :

$$\Phi_4 : \mathbf{x} \rightarrow (1, x_1, x_2, x_1^2, x_1x_2, x_2^2, x_1^3, x_1^2x_2, x_1x_2^2, x_2^3, x_1^4, x_1^3x_2, x_1^2x_2^2, x_1x_2^3, x_2^4)$$

3. What is the smallest value among the following choices that is \geq the VC dimension of a linear model in the transformed space?

- [a] 3
- [b] 5
- [c] 15
- [d] 20
- [e] 21

Gradient Descent

Consider the nonlinear error surface $E(u, v) = (ue^v - 2ve^{-u})^2$. We start at the point $(u, v) = (1, 1)$ and minimize this error using gradient descent in the uv space. Use $\eta = 0.1$ (learning rate, not step size).

4. What is the partial derivative of $E(u, v)$ with respect to u : $\frac{\partial E}{\partial u}$?
- [a] $(ue^v - 2ve^{-u})^2$
 - [b] $2(ue^v - 2ve^{-u})$
 - [c] $2(e^v + 2ve^{-u})$
 - [d] $2(e^v - 2ve^{-u})(ue^v - 2ve^{-u})$
 - [e] $2(e^v + 2ve^{-u})(ue^v - 2ve^{-u})$
5. How many iterations (among the given choices) does it take for the error $E(u, v)$ to fall below 10^{-14} for the first time? In your programs, make sure to use double precision to get the needed accuracy.
- [a] 1

- [b] 3
 [c] 5
 [d] 10
 [e] 17
6. After running enough iterations such that the error has just dropped below 10^{-14} , what are the closest values to the final (u, v) ?
- [a] (1.000, 1.000)
 [b] (0.713, 0.045)
 [c] (0.016, 0.112)
 [d] (-0.083, 0.029)
 [e] (0.045, 0.024)
7. Now, we will compare the performance of “coordinate descent.” In each iteration, we have two steps along the 2 coordinates. Step 1 is to move along the u coordinate only to reduce the error (assume first-order approximation holds like in gradient descent), and step 2 is to reevaluate and move along the v coordinate only to reduce the error (again, assume first-order approximation holds). Continue to use a learning rate of $\eta = 0.1$ as we did in gradient descent. What will the error $E(u, v)$ be closest to after 15 full iterations (30 steps)?
- [a] 10^{-1}
 [b] 10^{-7}
 [c] 10^{-14}
 [d] 10^{-17}
 [e] 10^{-20}

Logistic Regression

In this problem you will create your own target function f (probability in this case) and data set \mathcal{D} to see how Logistic Regression works. For simplicity, we will take f to be a 0/1 probability, so y is a deterministic function of \mathbf{x} .

Take $d = 2$ so you can visualize the problem, and choose a random line in the plane as the boundary between $f(\mathbf{x}) = 1$ (where y has to be +1) and $f(\mathbf{x}) = 0$ (where y has to be -1). Do this by taking two random uniformly distributed points on $[-1, 1] \times [-1, 1]$ and taking the line passing through them as the boundary between $y = \pm 1$. Take $\mathcal{X} = [-1, 1] \times [-1, 1]$ with uniform probability of picking each $\mathbf{x} \in \mathcal{X}$,

and take $N = 100$. Choose the inputs \mathbf{x}_n of the data set as random points in \mathcal{X} , and evaluate the output y_n for each \mathbf{x}_n .

Run Logistic Regression with Stochastic Gradient Descent to find g and estimate E_{out} (the **cross entropy** error) by generating a sufficiently large separate set of points to evaluating the error. Repeat the experiment 100 times with different targets and take the average. Initialize the weight vector of Logistic Regression to all zeros every time. Stop the algorithm when $\|\mathbf{w}^{(t-1)} - \mathbf{w}^{(t)}\| < 0.01$, where $\mathbf{w}^{(t)}$ denotes the weight vector at the end of epoch t (an epoch is one run through all the examples). Use a new random permutation of $1, 2, \dots, N$ to present the examples to the algorithm within each epoch, and use a learning rate of 0.01.

8. Which of the following is closest to E_{out} for $N = 100$?
 - [a] 0.03
 - [b] 0.05
 - [c] 0.07
 - [d] 0.09
 - [e] 0.11

9. How many epochs does it take on average for Logistic Regression to converge for $N = 100$ using the above initialization and termination rules, and specified learning rate? Pick the value that is closest to your results.
 - [a] 350
 - [b] 550
 - [c] 750
 - [d] 950
 - [e] 1750

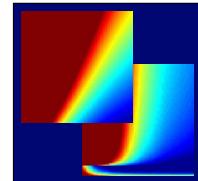
10. The Perceptron Learning Algorithm can be implemented as SGD using which of the following error functions $e_n(\mathbf{w})$:
 - [a] $e_n(\mathbf{w}) = e^{-y_n \mathbf{w}^\top \mathbf{x}_n}$
 - [b] $e_n(\mathbf{w}) = -y_n \mathbf{w}^\top \mathbf{x}_n$
 - [c] $e_n(\mathbf{w}) = (y_n - \mathbf{w}^\top \mathbf{x}_n)^2$
 - [d] $e_n(\mathbf{w}) = \ln(1 + e^{-y_n \mathbf{w}^\top \mathbf{x}_n})$
 - [e] $e_n(\mathbf{w}) = -\min(0, y_n \mathbf{w}^\top \mathbf{x}_n)$

Learning From Data

Yaser Abu-Mostafa, *Caltech*

<http://work.caltech.edu/telecourse>

Self-paced version



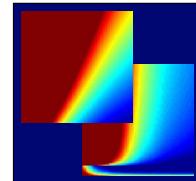
Answer Key To Homework # 5

1. [c]
2. [d]
3. [c]
4. [e]
5. [d]
6. [e]
7. [a]
8. [d]
9. [a]
10. [e]

Visit the forum (<http://book.caltech.edu/bookforum>) for discussion. Please follow the forum guidelines for posting answers (see the “BEFORE posting answers” announcement at the top thread there).

Learning From Data*Caltech*<http://work.caltech.edu/telecourse.html>

2012



Online Homework # 6

Collaboration in the sense of discussions is allowed, but you should NOT discuss your selected answers with anyone. Books and notes can be consulted. All questions will have multiple choice answers, the choices being labeled [a], [b], [c], [d], [e]. Most questions will have 5 possible choices, although some may have fewer choices. You should enter your solutions online by logging into your account at the course web site.

Note about the homeworks

- The goal of the homeworks is to facilitate a deeper understanding of the course material. The questions are not designed to be puzzles with catchy answers. They are meant to make you roll your sleeves, face uncertainties, and approach the problem from different angles.
- The problems range from easy to hard, and from theoretical to practical. Some problems require running a full experiment to arrive at the answer.
- The answer may not be obvious or numerically close to one of the choices. The intent is to prompt discussion and exchange of ideas.
- Speaking of discussion, you are encouraged to take part in the forum

<http://book.caltech.edu/bookforum>

where there are many threads about each homework. We hope that you will contribute to the discussion as well.

- Please follow the forum guidelines for posting answers (see the “BEFORE posting answers” top thread there).

Overfitting and Deterministic Noise

1. Deterministic noise depends on \mathcal{H} , as some models approximate f better than others. Assume $\mathcal{H}' \subset \mathcal{H}$ and that f is fixed. In general, if we use \mathcal{H}' instead of \mathcal{H} , how does deterministic noise behave?
 - [a] In general, deterministic noise will decrease.
 - [b] In general, deterministic noise will increase.
 - [c] In general, deterministic noise will be the same.
 - [d] There is no trend.

Overfitting and Regularization With Weight Decay

In the following problems use the data provided in the files

<http://work.caltech.edu/data/in.dta>

<http://work.caltech.edu/data/out.dta>

as a training and test set respectively. Each line of the files corresponds to a two-dimensional input $\mathbf{x} = (x_1, x_2)$, so that $\mathcal{X} = \mathbb{R}^2$, followed by the corresponding label from $\mathcal{Y} = \{-1, 1\}$. We are going to apply Linear Regression with a non-linear transformation for classification. The nonlinear transformation is given by

$$\phi(x_1, x_2) = (1, x_1, x_2, x_1^2, x_2^2, x_1 x_2, |x_1 - x_2|, |x_1 + x_2|).$$

Recall that the classification error is defined as the fraction of miss-classified points.

2. Run Linear Regression on the training set after performing the non-linear transformation. What values are closest to the in-sample and out-of-sample classification errors, respectively?
 - [a] 0.03, 0.08
 - [b] 0.03, 0.10
 - [c] 0.04, 0.09
 - [d] 0.04, 0.11
 - [e] 0.05, 0.10
3. Now add weight decay to Linear Regression, that is add the term $\frac{\lambda}{N} \sum_{i=0}^7 w_i^2$ to the squared in-sample error, using $\lambda = 10^k$. What are the closest values to the in-sample and out-of-sample classification errors, respectively, for $k = -3$? Recall that the solution for Linear Regression with Weight Decay was derived in class.
 - [a] 0.01, 0.02

- [b] 0.02, 0.04
 [c] 0.02, 0.06
 [d] 0.03, 0.08
 [e] 0.03, 0.10
4. Now use $k = 3$. What are the closest values to the new in-sample and out-of-sample classification errors, respectively?
- [a] 0.2, 0.2
 [b] 0.2, 0.3
 [c] 0.3, 0.3
 [d] 0.3, 0.4
 [e] 0.4, 0.4
5. What value of k , among the following choices, achieves the smallest out-of-sample classification error?
- [a] 2
 [b] 1
 [c] 0
 [d] -1
 [e] -2
6. What value is closest to the minimum out-of-sample classification error achieved by varying k (limiting k to integer values)?
- [a] 0.04
 [b] 0.06
 [c] 0.08
 [d] 0.10
 [e] 0.12

Regularization With Polynomials

Polynomial models can be viewed as linear models in a space \mathcal{Z} , under a nonlinear transform $\Phi : \mathcal{X} \rightarrow \mathcal{Z}$. Here, Φ transforms the scalar x into a vector \mathbf{z} of Legendre polynomials, $\mathbf{z} = (1, L_1(x), L_2(x), \dots, L_Q(x))$. Our hypothesis set will be expressed as a linear combination of these polynomials,

$$\mathcal{H}_Q = \left\{ h \mid h(x) = \mathbf{w}^T \mathbf{z} = \sum_{q=0}^Q w_q L_q(x) \right\},$$

where $L_0(x) = 1$.

- 7.** Consider the following hypothesis set defined by the constraint:

$$\mathcal{H}(Q, C, Q_o) = \{h \mid h(x) = \mathbf{w}^T \mathbf{z} \in \mathcal{H}_Q; w_q = C \text{ for } q \geq Q_o\},$$

which of the following statements is correct:

- [a] $\mathcal{H}(10, 0, 3) \cup \mathcal{H}(10, 0, 4) = \mathcal{H}_4$
- [b] $\mathcal{H}(10, 1, 3) \cup \mathcal{H}(10, 1, 4) = \mathcal{H}_3$
- [c] $\mathcal{H}(10, 0, 3) \cap \mathcal{H}(10, 0, 4) = \mathcal{H}_2$
- [d] $\mathcal{H}(10, 1, 3) \cap \mathcal{H}(10, 1, 4) = \mathcal{H}_1$
- [e] None of the above

Neural Networks

- 8.** A fully connected Neural Network has $L = 2$; $d^{(0)} = 5$, $d^{(1)} = 3$, $d^{(2)} = 1$. If only products of the form $w_{ij}^{(l)} x_i^{(l-1)}$, $w_{ij}^{(l)} \delta_j^{(l)}$, and $x_i^{(l-1)} \delta_j^{(l)}$ count as operations (without counting anything else), which of the following is the closest to the total number of operations required in a single iteration of backpropagation (using SGD on one data point)?

- [a] 30
- [b] 35
- [c] 40
- [d] 45
- [e] 50

A Neural Network has 10 input units (including $x_0^{(0)}$, the constant value), one output unit, and 36 hidden units (including $x_0^{(l)}$, the constant inputs of each hidden layer). The hidden units can be arranged in any number of layers $l = 1, \dots, L - 1$, and each layer is fully connected to the layer above it.

- 9.** What is the minimum possible number of weights that such a network can have?

- [a] 46
- [b] 47
- [c] 56
- [d] 57

[e] 58

10. What is the maximum possible number of weights that such a network can have?

[a] 386

[b] 493

[c] 494

[d] 509

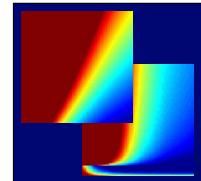
[e] 510

Learning From Data

Caltech

<http://work.caltech.edu/telecourse.html>

2012



Answer Key To Online Homework # 6

1. [b]
2. [a]
3. [d]
4. [e]
5. [d]
6. [b]
7. [c]
8. [d]
9. [a]
10. [e]

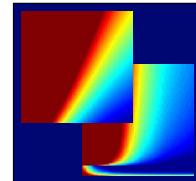
Visit the forum (<http://book.caltech.edu/bookforum>) for discussion. Please follow the forum guidelines for posting answers (see the “BEFORE posting answers” top thread there).

Learning From Data

Caltech

<http://work.caltech.edu/telecourse.html>

2012



Online Homework # 7

Collaboration in the sense of discussions is allowed, but you should NOT discuss your selected answers with anyone. Books and notes can be consulted. All questions will have multiple choice answers, the choices being labeled [a], [b], [c], [d], [e]. Most questions will have 5 possible choices, although some may have fewer choices. You should enter your solutions online by logging into your account at the course web site.

Note about the homeworks

- The goal of the homeworks is to facilitate a deeper understanding of the course material. The questions are not designed to be puzzles with catchy answers. They are meant to make you roll your sleeves, face uncertainties, and approach the problem from different angles.
- The problems range from easy to hard, and from theoretical to practical. Some problems require running a full experiment to arrive at the answer.
- The answer may not be obvious or numerically close to one of the choices. The intent is to prompt discussion and exchange of ideas.
- Speaking of discussion, you are encouraged to take part in the forum

<http://book.caltech.edu/bookforum>

where there are many threads about each homework. We hope that you will contribute to the discussion as well.

- Please follow the forum guidelines for posting answers (see the “BEFORE posting answers” top thread there).

Validation

In the following problems, use the data provided in the files `in.dta` and `out.dta` for Homework #6. We are going to apply linear regression with a nonlinear transformation for classification (without regularization). The nonlinear transformation is given by ϕ_0 through ϕ_7 which transform (x_1, x_2) into

$$1 \quad x_1 \quad x_2 \quad x_1^2 \quad x_2^2 \quad x_1 x_2 \quad |x_1 - x_2| \quad |x_1 + x_2|$$

To illustrate how taking out points for validation affects the performance, we will consider the hypotheses trained on $\mathcal{D}_{\text{train}}$ (without restoring the full \mathcal{D} for training after validation is done).

1. Split `in.dta` into training (first 25 examples) and validation (last 10 examples). Train on the 25 examples only, using the validation set of 10 examples to select between six models that apply linear regression to ϕ_0 through ϕ_k , with $k = 3, 4, 5, 6, 7$. For which model is the classification error on the validation set smallest?

- [a] $k = 3$
- [b] $k = 4$
- [c] $k = 5$
- [d] $k = 6$
- [e] $k = 7$

2. Evaluate the out-of-sample classification error using `out.dta` on the 5 models to see how well the validation set predicted the best of the 5 models. For which model is the out-of-sample classification error smallest?

- [a] $k = 3$
- [b] $k = 4$
- [c] $k = 5$
- [d] $k = 6$
- [e] $k = 7$

3. Reverse the role of training and validation sets; now training with the 10 examples and validating with the 25 examples. For which model is the classification error on the validation set smallest?

- [a] $k = 3$
- [b] $k = 4$

- [c] $k = 5$
 - [d] $k = 6$
 - [e] $k = 7$
4. Once again evaluate the out-of-sample classification error using `out.dta` on the 5 models to see how well the validation set predicted the best of the 5 models. For which model is the out-of-sample classification error smallest?
- [a] $k = 3$
 - [b] $k = 4$
 - [c] $k = 5$
 - [d] $k = 6$
 - [e] $k = 7$
5. What values are closest to the out-of-sample classification error obtained for the model chosen in each of the above two experiments, respectively?
- [a] 0.0, 0.1
 - [b] 0.1, 0.2
 - [c] 0.1, 0.3
 - [d] 0.2, 0.2
 - [e] 0.2, 0.3

Estimators

6. Let e_1 and e_2 be independent random variables, distributed uniformly over the interval $[0, 1]$. Let $e = \min(e_1, e_2)$. The expected values of e_1, e_2, e are closest to
- [a] 0.5, 0.5, 0
 - [b] 0.5, 0.5, 0.1
 - [c] 0.5, 0.5, 0.25
 - [d] 0.5, 0.5, 0.4
 - [e] 0.5, 0.5, 0.5

Cross Validation

7. You are given the data points: $(-1, 0), (\rho, 1), (1, 0)$, $\rho \geq 0$, and a choice between two models: constant [$h_0(x) = b$] and linear [$h_1(x) = ax + b$]. For which value of ρ would the two models be tied using leave-one-out cross-validation with the squared error measure?
- [a] $\sqrt{\sqrt{3} + 4}$
 - [b] $\sqrt{\sqrt{3} - 1}$
 - [c] $\sqrt{9 + 4\sqrt{66}}$
 - [d] $\sqrt{9 - \sqrt{66}}$
 - [e] None of the above

PLA vs. SVM

In the following problems, we compare PLA to SVM with hard margin on linearly separable data sets. For each run, you will create your own target function f and data set \mathcal{D} . Take $d = 2$ and choose a random line in the plane as your target function f (do this by taking two random, uniformly distributed points on $[-1, 1] \times [-1, 1]$ and taking the line passing through them), where one side of the line maps to $+1$ and the other maps to -1 . Choose the inputs \mathbf{x}_n of the data set as random points in $\mathcal{X} = [-1, 1] \times [-1, 1]$, and evaluate the target function on each \mathbf{x}_n to get the corresponding output y_n . If all data points are on one side of the line, discard the run and start a new run.

Start PLA with the all-zero vector and pick the misclassified point for each PLA iteration at random. Run PLA to find the final hypothesis g_{PLA} and measure the difference between f and g_{PLA} as $\mathbb{P}[f(\mathbf{x}) \neq g_{\text{PLA}}(\mathbf{x})]$ (you can either calculate this exactly, or approximate it by generating a sufficiently large separate set of points to evaluate it). Now, run SVM on the same data to find the final hypothesis g_{SVM} by solving

$$\begin{aligned} \min_{\mathbf{w}, b} \quad & \frac{1}{2} \mathbf{w}^T \mathbf{w} \\ \text{s.t.} \quad & y_n (\mathbf{w}^T \mathbf{x}_n + b) \geq 1 \end{aligned}$$

using quadratic programming on the primal or the dual problem. Measure the difference between f and g_{SVM} as $\mathbb{P}[f(\mathbf{x}) \neq g_{\text{SVM}}(\mathbf{x})]$, and count the number of support vectors you get in each run.

8. For $N = 10$, repeat the above experiment for 1000 runs. How often is g_{SVM} better than g_{PLA} in approximating f ? The percentage of time is closest to:

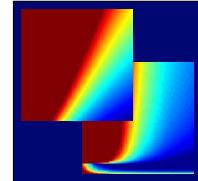
- [a] 20%
 - [b] 40%
 - [c] 60%
 - [d] 80%
 - [e] 100%
9. For $N = 100$, repeat the above experiment for 1000 runs. How often is g_{SVM} better than g_{PLA} in approximating f ? The percentage of time is closest to:
- [a] 10%
 - [b] 30%
 - [c] 50%
 - [d] 70%
 - [e] 90%
10. For the case $N = 100$, which of the following is the closest to the average number of support vectors of g_{SVM} (averaged over the 1000 runs)?
- [a] 2
 - [b] 3
 - [c] 5
 - [d] 10
 - [e] 20

Learning From Data

Caltech

<http://work.caltech.edu/telecourse.html>

2012



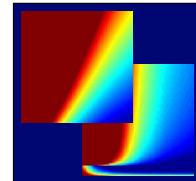
Answer Key To Online Homework # 7

1. [d]
2. [e]
3. [d]
4. [d]
5. [b]
6. [d]
7. [c] (typo, so [e] is accepted)
8. [c]
9. [d]
10. [b]

Visit the forum (<http://book.caltech.edu/bookforum>) for discussion. Please follow the forum guidelines for posting answers (see the “BEFORE posting answers” top thread there).

Learning From Data*Caltech*<http://work.caltech.edu/telecourse.html>

2012



Online Homework # 8

Collaboration in the sense of discussions is allowed, but you should NOT discuss your selected answers with anyone. Books and notes can be consulted. All questions will have multiple choice answers, the choices being labeled [a], [b], [c], [d], [e]. Most questions will have 5 possible choices, although some may have fewer choices. You should enter your solutions online by logging into your account at the course web site.

Note about the homeworks

- The goal of the homeworks is to facilitate a deeper understanding of the course material. The questions are not designed to be puzzles with catchy answers. They are meant to make you roll your sleeves, face uncertainties, and approach the problem from different angles.
- The problems range from easy to hard, and from theoretical to practical. Some problems require running a full experiment to arrive at the answer.
- The answer may not be obvious or numerically close to one of the choices. The intent is to prompt discussion and exchange of ideas.
- Speaking of discussion, you are encouraged to take part in the forum

<http://book.caltech.edu/bookforum>

where there are many threads about each homework. We hope that you will contribute to the discussion as well.

- Please follow the forum guidelines for posting answers (see the “BEFORE posting answers” top thread there).

Primal versus Dual Problem

1. Recall that N is the size of the data set and d is the dimensionality of the input space. The original formulation of the hard-margin SVM problem (minimize $\frac{1}{2}\mathbf{w}^T\mathbf{w}$ subject to the inequality constraints), without going through the Lagrangian dual problem, is

- [a] a quadratic programming problem with N variables
- [b] a quadratic programming problem with $N + 1$ variables
- [c] a quadratic programming problem with d variables
- [d] a quadratic programming problem with $d + 1$ variables
- [e] not a quadratic programming problem

Support Vector Machines With Soft Margins

In this homework set, we are going to experiment with a real-world dataset. Download the processed US Postal Service Zip Code dataset with extracted features of symmetry and intensity for training and testing:

<http://www.amlbook.com/data/zip/features.train>

<http://www.amlbook.com/data/zip/features.test>

(the format of each row is: **digit symmetry intensity**). We will train two types of binary classifiers; one-versus-one (one digit is class +1 and another digit is class -1, with the rest of the digits disregarded), and one-versus-all (one digit is class +1 and the rest of the digits are class -1).

The data set has thousands of points, and some quadratic programming packages cannot handle this size. We recommend that you use the packages in libsvm:

<http://www.csie.ntu.edu.tw/~cjlin/libsvm/>

Implement SVM with soft margin on the above zip-code data set by solving

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2} \sum_{n=1}^N \sum_{m=1}^N \alpha_n \alpha_m y_n y_m K(\mathbf{x}_n, \mathbf{x}_m) - \sum_{n=1}^N \alpha_n \\ \text{s.t.} \quad & \sum_{n=1}^N y_n \alpha_n = 0 \\ & 0 \leq \alpha_n \leq C \quad n = 1, \dots, N \end{aligned}$$

When evaluating E_{in} and E_{out} of the resulting classifier, use binary classification error.

Practical remarks:

- (i) For the purpose of this homework, do not scale the data when you use libsvm or other packages, lest you should change the effective kernel and get different results.
- (ii) In some packages, you need to specify double precision.
- (iii) In 10-fold cross validation, if the data size is not a multiple of 10, the sizes of the 10 subsets may be off by 1 data point.

Polynomial Kernels

Consider the polynomial kernel $K(\mathbf{x}_n, \mathbf{x}_m) = (1 + \mathbf{x}_n^T \mathbf{x}_m)^Q$, where Q is the degree of the polynomial.

2. With $C = 0.01$ and $Q = 2$, which of the following classifiers has the **highest** E_{in} ?
 - [a] 0 versus all
 - [b] 2 versus all
 - [c] 4 versus all
 - [d] 6 versus all
 - [e] 8 versus all
3. With $C = 0.01$ and $Q = 2$, which of the following classifiers has the **lowest** E_{in} ?
 - [a] 1 versus all
 - [b] 3 versus all
 - [c] 5 versus all
 - [d] 7 versus all
 - [e] 9 versus all
4. Comparing the two classifiers from Problems 2 and 3, which of the following values is the closest to the difference between the number of support vectors of these two classifiers?
 - [a] 600
 - [b] 1200
 - [c] 1800
 - [d] 2400
 - [e] 3000

5. Consider the 1 versus 5 classifier with $Q = 2$, and apply ten-fold cross validation to pick C (among the given choices). Which of the following values of C results in the lowest E_{cv} ?
- [a] $C = 0.01$
 - [b] $C = 0.1$
 - [c] $C = 1$
 - [d] $C = 10$
 - [e] $C = 100$
6. Comparing the choices given in Problem 5, which of the following statements is correct?
- [a] E_{in} is inversely proportional to C .
 - [b] E_{out} goes down when C goes up.
 - [c] E_{cv} goes down when C goes up.
 - [d] The number of support vectors goes down when C goes up.
 - [e] The number of support vectors goes up when C goes up.
7. In the 1 versus 5 classifier, comparing $Q = 2$ with $Q = 5$, which of the following statements is correct?
- [a] When $C = 0.01$, E_{in} is higher at $Q = 5$.
 - [b] When $C = 1$, the number of support vectors is lower at $Q = 5$.
 - [c] When $C = 100$, E_{in} is lower at $Q = 5$.
 - [d] When $C = 10^4$, E_{out} is lower at $Q = 5$.
 - [e] When $C = 10^6$, E_{out} is lower at $Q = 5$.

RBF Kernel

Consider the radial basis function (RBF) kernel $K(\mathbf{x}_n, \mathbf{x}_m) = \exp(-\|\mathbf{x}_n - \mathbf{x}_m\|^2)$. Focus on the 1 versus 5 classifier.

8. Which of the following values of C results in the lowest E_{in} ?
- [a] $C = 0.01$
 - [b] $C = 1$
 - [c] $C = 100$
 - [d] $C = 10^4$

[e] $C = 10^6$

9. Which of the following values of C results in the lowest E_{out} ?

[a] $C = 0.01$

[b] $C = 1$

[c] $C = 100$

[d] $C = 10^4$

[e] $C = 10^6$

10. Applying ten-fold cross validation, which of the following values of C has the lowest E_{cv} ?

[a] $C = 1$

[b] $C = 100$

[c] $C = 10^4$

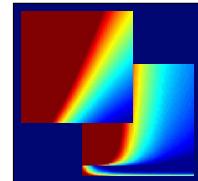
[d] $C = 10^6$

Learning From Data

Yaser Abu-Mostafa, *Caltech*

<http://work.caltech.edu/telecourse>

Self-paced version



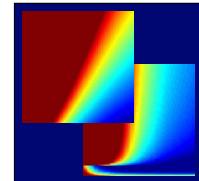
Answer Key To Homework # 8

1. [d]
2. [a]
3. [a]
4. [c]
5. [d]
6. [b]
7. [b]
8. [c]
9. [e]
10. [c]

Visit the forum (<http://book.caltech.edu/bookforum>) for discussion. Please follow the forum guidelines for posting answers (see the “BEFORE posting answers” announcement at the top thread there).

Learning From Data*Caltech*<http://work.caltech.edu/telecourse.html>

2012



Online Final

Collaboration in the sense of discussions is allowed, but you should NOT discuss your selected answers with anyone. Books and notes can be consulted. All questions will have multiple choice answers, the choices being labeled [a], [b], [c], [d], [e]. Most questions will have 5 possible choices, although some may have fewer choices. You should enter your solutions online by logging into your account at the course web site.

Nonlinear transforms

1. The polynomial transform of order $Q = 10$ applied to \mathcal{X} of dimension $d = 2$ results in a \mathcal{Z} space of what dimensionality (not counting the constant coordinate $x_0 = 1$ or $z_0 = 1$)?

- [a] 12
- [b] 20
- [c] 35
- [d] 100
- [e] None of the above

Bias and Variance

2. Recall that the average hypothesis \bar{g} was based on training the same model \mathcal{H} on different data sets \mathcal{D} to get $g^{(\mathcal{D})} \in \mathcal{H}$, and taking the expected value of $g^{(\mathcal{D})}$ w.r.t. \mathcal{D} to get \bar{g} . Which of the following models \mathcal{H} could result in $\bar{g} \notin \mathcal{H}$?

- [a] A singleton \mathcal{H} (\mathcal{H} has one hypothesis)
- [b] \mathcal{H} is the set of constant, real-valued hypotheses

- [c] \mathcal{H} is the linear regression model
- [d] \mathcal{H} is the logistic regression model
- [e] None of the above

Overfitting

3. Which of the following statements is false
 - [a] If there is overfitting, there must be two or more hypotheses that have different values of E_{in}
 - [b] If there is overfitting, there must be two or more hypotheses that have different values of E_{out}
 - [c] If there is overfitting, there must be two or more hypotheses that have different values of $(E_{\text{out}} - E_{\text{in}})$
 - [d] We can determine if there is overfitting by comparing the values of $(E_{\text{out}} - E_{\text{in}})$
 - [e] We cannot determine overfitting based on one hypothesis only

4. Which of the following statements is true

- [a] Deterministic noise cannot occur with stochastic noise
- [b] Deterministic noise does not depend on the learning model
- [c] Deterministic noise does not depend on the target function
- [d] Stochastic noise does not depend on the learning model
- [e] Stochastic noise does not depend on the target distribution

Regularization

5. The regularized weight \mathbf{w}_{reg} is a solution to:

$$\text{minimize } \frac{1}{N} \sum_{n=1}^N (\mathbf{w}^T \mathbf{x}_n - y_n)^2 \text{ subject to } \mathbf{w}^T \Gamma^T \Gamma \mathbf{w} \leq C,$$

where Γ is a matrix. If $\mathbf{w}_{\text{lin}}^T \Gamma^T \Gamma \mathbf{w}_{\text{lin}} \leq C$, where \mathbf{w}_{lin} is the linear regression solution, then what is \mathbf{w}_{reg} ?

- [a] $\mathbf{w}_{\text{reg}} = \mathbf{w}_{\text{lin}}$
- [b] $\mathbf{w}_{\text{reg}} = \Gamma \mathbf{w}_{\text{lin}}$

[c] $\mathbf{w}_{\text{reg}} = \Gamma^T \Gamma \mathbf{w}_{\text{lin}}$

[d] $\mathbf{w}_{\text{reg}} = C \Gamma \mathbf{w}_{\text{lin}}$

[e] $\mathbf{w}_{\text{reg}} = C \mathbf{w}_{\text{lin}}$

6. Soft-order constraints that regularize polynomial models can be

[a] written as hard-order constraints

[b] translated into augmented error

[c] determined from the value of the VC dimension

[d] used to decrease both E_{in} and E_{out}

[e] None of the above is true

Regularized Linear Regression

We are going to experiment with linear regression for classification on a real world dataset. Download the processed US Postal Service Zip Code dataset with extracted features of symmetry and intensity for training and testing:

<http://www.amlbook.com/data/zip/features.train>

<http://www.amlbook.com/data/zip/features.test>

(the format of each row is: **digit symmetry intensity**). We will train two types of binary classifiers; one-versus-one (one digit is class +1 and another digit is class -1, with the rest of the digits disregarded), and one-versus-all (one digit is class +1 and the rest of the digits are class -1). When evaluating E_{in} and E_{out} , use binary classification error. Implement the regularized least-squares linear regression for classification that minimizes

$$\frac{1}{N} \sum_{n=1}^N (\mathbf{w}^T \mathbf{z}_n - y_n)^2 + \frac{\lambda}{N} \mathbf{w}^T \mathbf{w}$$

where \mathbf{w} includes w_0 .

7. Set $\lambda = 1$ and do not apply a feature transform (i.e., use $\mathbf{z} = \mathbf{x}$). Which among the following classifiers has the lowest E_{in} ?

[a] 5 versus all

[b] 6 versus all

[c] 7 versus all

[d] 8 versus all

[e] 9 versus all

8. Now, apply a feature transform $\mathbf{z} = (1, x_1, x_2, x_1x_2, x_1^2, x_2^2)$, and set $\lambda = 1$. Which among the following classifiers has the lowest E_{out} ?
- [a] 0 versus all
 - [b] 1 versus all
 - [c] 2 versus all
 - [d] 3 versus all
 - [e] 4 versus all
9. Comparing the two cases (without transform versus with transform), which of the following statements is correct for $\lambda = 1$ considering ‘0 versus all’ through ‘9 versus all’? Percentages here are relative, not absolute.
- [a] Overfitting always occurs when we use the transform
 - [b] The transform always improves the out-of-sample performance by at least 5%
 - [c] The transform does not make any difference in the out-of-sample performance
 - [d] The transform always worsens the out-of-sample performance by at least 5%
 - [e] The transform improves the out-of-sample performance of ‘5 versus all’ by less than 5%
10. Train the ‘1 versus 5’ classifier with $\mathbf{z} = (1, x_1, x_2, x_1x_2, x_1^2, x_2^2)$ with $\lambda = 0.01$ and $\lambda = 1$. Which of the following statements is correct?
- [a] Overfitting occurs (from $\lambda = 1$ to $\lambda = 0.01$)
 - [b] The two classifiers have the same E_{in}
 - [c] The two classifiers have the same E_{out}
 - [d] When λ goes up, both E_{in} and E_{out} go up
 - [e] When λ goes up, both E_{in} and E_{out} go down

Aggregation

11. Given two learned hypotheses g_1 and g_2 , we take their average as the aggregate hypothesis g . If we use mean-squared error, which of the following statements is true?
- [a] $E_{\text{out}}(g)$ cannot be worse than $E_{\text{out}}(g_1)$

- [b] $E_{\text{out}}(g)$ cannot be worse than the smaller of $E_{\text{out}}(g_1)$ and $E_{\text{out}}(g_2)$
- [c] $E_{\text{out}}(g)$ cannot be worse than the average of $E_{\text{out}}(g_1)$ and $E_{\text{out}}(g_2)$
- [d] $E_{\text{out}}(g)$ has to be between $E_{\text{out}}(g_1)$ and $E_{\text{out}}(g_2)$
- [e] None of the above

Support Vector Machines

- 12.** Consider the following training set generated from a target function $f : \mathcal{X} \rightarrow \{-1, +1\}$ where $\mathcal{X} = \mathbb{R}^2$

$$\begin{array}{lll} \mathbf{x}_1 = (1, 0), y_1 = -1 & \mathbf{x}_2 = (0, 1), y_2 = -1 & \mathbf{x}_3 = (0, -1), y_3 = -1 \\ \mathbf{x}_4 = (-1, 0), y_4 = +1 & \mathbf{x}_5 = (0, 2), y_5 = +1 & \mathbf{x}_6 = (0, -2), y_6 = +1 \\ \mathbf{x}_7 = (-2, 0), y_7 = +1 & & \end{array}$$

Transform this training set into another two-dimensional space \mathcal{Z}

$$z_1 = x_2^2 - 2x_1 - 1 \quad z_2 = x_1^2 - 2x_2 + 1$$

Using geometry, what values of \mathbf{w} (without w_0) and b specify the separating plane $\mathbf{w}^T \mathbf{z} + b = 0$ in the \mathcal{Z} space that maximizes the margin? The values of w_1, w_2, b are:

- [a] $-1, 1, -0.5$
 - [b] $1, -1, -0.5$
 - [c] $1, 0, -0.5$
 - [d] $0, 1, -0.5$
 - [e] None of the above
- 13.** Consider the same training set of the previous problem, but instead of explicitly transforming the input space \mathcal{X} , apply the SVM algorithm with the kernel

$$K(\mathbf{x}, \mathbf{x}') = (1 + \mathbf{x}^T \mathbf{x}')^2$$

(which corresponds to a second-order polynomial transformation). Set up the expression for $\mathcal{L}(\alpha_1 \dots \alpha_7)$ and solve for the optimal $\alpha_1, \dots, \alpha_7$ (numerically, using a quadratic programming package). How many support vectors do you get?

- [a] 3
- [b] 4
- [c] 5

[d] 6

[e] 7

Radial Basis Functions

We experiment with the RBF model, both in regular form (Lloyd + pseudo-inverse) with K centers:

$$\text{sign} \left(\sum_{k=1}^K w_k \exp(-\gamma \|\mathbf{x} - \mu_k\|^2) + b \right)$$

(notice that there is a bias term), and in kernel form (using the RBF kernel in hard-margin SVM):

$$\text{sign} \left(\sum_{\alpha_n > 0} \alpha_n y_n \exp(-\gamma \|\mathbf{x} - \mathbf{x}_n\|^2) + b \right).$$

The input space is $\mathcal{X} = [-1, 1] \times [-1, 1]$ with uniform probability distribution, and the target is

$$f(\mathbf{x}) = \text{sign}(x_2 - x_1 + 0.25 \sin(\pi x_1))$$

which is slightly nonlinear in the \mathcal{X} space. In each run, generate 100 training points at random using this target, and apply both forms of RBF to these training points. Here are some guidelines:

- Repeat the experiment for as many runs as needed to get the answer to be stable (statistically away from flipping to the closest competing answer).
- In case a data set is not linearly separable in the ' \mathcal{Z} space' by the RBF kernel using hard-margin SVM, discard the run but keep track of how often this happens.
- When you use Lloyd's algorithm, initialize the centers to random points in \mathcal{X} and iterate until there is no change from iteration to iteration. If a cluster becomes empty, discard the run and repeat.

- 14.** For $\gamma = 1.5$, how often do you get a data set that is not linearly separable by the RBF kernel (using hard-margin SVM). *Hint: Run the usual hard-margin SVM, then check that the solution has $E_{\text{in}} = 0$.*

[a] $\leq 5\%$ of the time

[b] $> 5\%$ but $\leq 10\%$ of the time

[c] $> 10\%$ but $\leq 20\%$ of the time

[d] $> 20\%$ but $\leq 40\%$ of the time

[e] $> 40\%$ of the time

15. If we use $K = 9$ for regular RBF and take $\gamma = 1.5$, how often does the kernel form beat the regular form (after discarding the runs mentioned above, if any) in terms of E_{out} ?
- [a] $\leq 15\%$ of the time
 - [b] $> 15\%$ but $\leq 30\%$ of the time
 - [c] $> 30\%$ but $\leq 50\%$ of the time
 - [d] $> 50\%$ but $\leq 75\%$ of the time
 - [e] $> 75\%$ of the time
16. If we use $K = 12$ for regular RBF and take $\gamma = 1.5$, how often does the kernel form beat the regular form (after discarding the runs mentioned above, if any) in terms of E_{out} ?
- [a] $\leq 10\%$ of the time
 - [b] $> 10\%$ but $\leq 30\%$ of the time
 - [c] $> 30\%$ but $\leq 60\%$ of the time
 - [d] $> 60\%$ but $\leq 90\%$ of the time
 - [e] $> 90\%$ of the time
17. Comparing $K = 9$ to $K = 12$ (respectively) for regular RBF, with $\gamma = 1.5$, which of the following statements is true
- [a] E_{in} goes down but E_{out} goes up
 - [b] E_{in} goes up but E_{out} goes down
 - [c] Both E_{in} and E_{out} go up
 - [d] Both E_{in} and E_{out} go down
 - [e] There is no change
18. Comparing $\gamma = 1.5$ to $\gamma = 2$ (respectively) for regular RBF, with $K = 9$, which of the following statements is true
- [a] E_{in} goes down but E_{out} goes up
 - [b] E_{in} goes up but E_{out} goes down
 - [c] Both E_{in} and E_{out} go up
 - [d] Both E_{in} and E_{out} go down
 - [e] There is no change

19. What is the percentage of time that regular RBF achieves $E_{\text{in}} = 0$ with $K = 9$ and $\gamma = 1.5$?

- [a] $\leq 10\%$ of the time
- [b] $> 10\%$ but $\leq 20\%$ of the time
- [c] $> 20\%$ but $\leq 30\%$ of the time
- [d] $> 30\%$ but $\leq 50\%$ of the time
- [e] $> 50\%$ of the time

Bayesian Priors

20. Let $f \in [0, 1]$ be the unknown probability of getting a heart attack for people in a certain population. Notice that f is just a constant, not a function, for simplicity. We want to model f using a hypothesis $h \in [0, 1]$. Before we see any data, we assume that $P(h = f)$ is uniform over $h \in [0, 1]$ (the prior). We pick one sample from the population, and it turns out that they had a heart attack. Which of the following is true about the posterior probability that $h = f$ given this sample point?

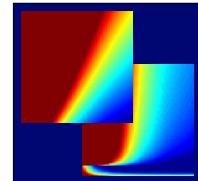
- [a] The posterior is uniform over $[0, 1]$
- [b] The posterior increases linearly over $[0, 1]$
- [c] The posterior increases nonlinearly over $[0, 1]$
- [d] The posterior is a delta function at 1 (implying f has to be 1)
- [e] The posterior cannot be evaluated based on the given information

Learning From Data

Caltech

<http://work.caltech.edu/telecourse.html>

2012



Answer Key To Online Final

- | | |
|---------|---------|
| 1. [e] | 11. [c] |
| 2. [d] | 12. [c] |
| 3. [d] | 13. [c] |
| 4. [d] | 14. [a] |
| 5. [a] | 15. [e] |
| 6. [b] | 16. [d] |
| 7. [d] | 17. [d] |
| 8. [b] | 18. [c] |
| 9. [e] | 19. [a] |
| 10. [a] | 20. [b] |

Visit the forum (<http://book.caltech.edu/bookforum>) for discussion. Please follow the forum guidelines for posting answers (see the “BEFORE posting answers” top thread there).