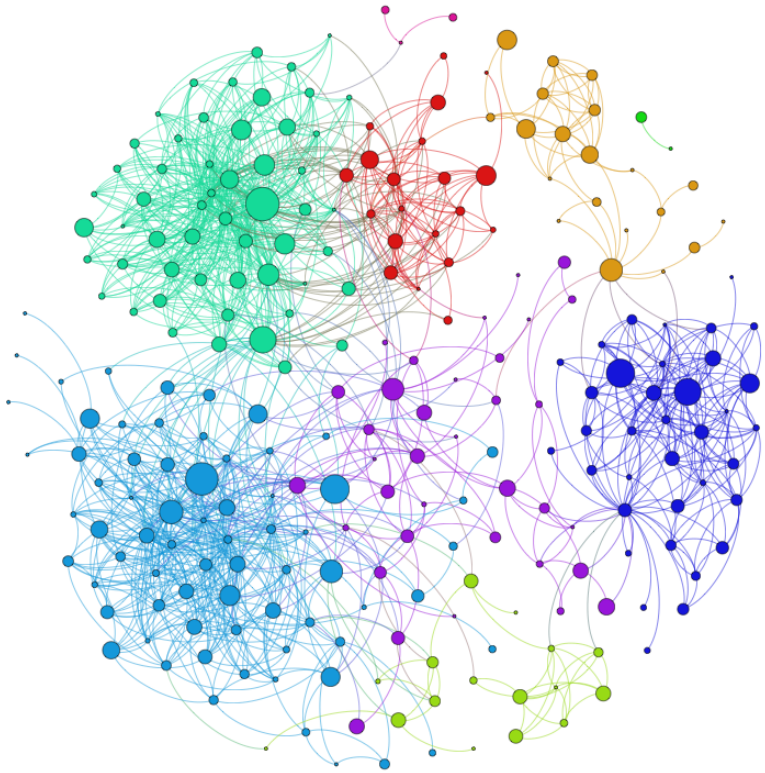
Foundations of Statistical Learning

Jiaming Mao

Xiamen University

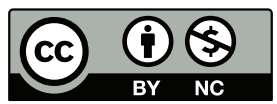


Copyright © 2017–2021, by Jiaming Mao

This version: Spring 2021

Contact: jmao@xmu.edu.cn

Course homepage: jiamingmao.github.io/data-analysis



All materials are licensed under the Creative Commons Attribution-NonCommercial 4.0 International License.

名人名言

"All models are wrong but some are useful." – George Box

"The existence of a problem in knowledge depends on the future being different from the past, while the possibility of a solution of the problem depends on the future being like the past." – Frank Knight

© Jiaming Mao

The Learning Problem

- Given variables x and y, suppose we are interested in predicting the value of y based on the value of x.
  - x: **feature**; **input**; **predictor; independent variable**
  - y: **target**; **output**; **response; dependent variable**
- For simplicity, assume there exists a function f such that $y = f(x)$[1]. f is the **target function** that we want to learn: to predict the value of y is to learn f[2].

[1] i.e., given x, y is completely determined.

[2] In the statistics and econometrics literature, **learning** is called **estimation**. In this lecture, we use the two terms interchangeably.

© Jiaming Mao

The Learning Problem

- Let the observed data be $\mathcal{D} = \{(x_1, y_1), \ldots, (x_N, y_N)\}$.
- Start with a set of candidate **hypotheses** which you think are likely to represent f:

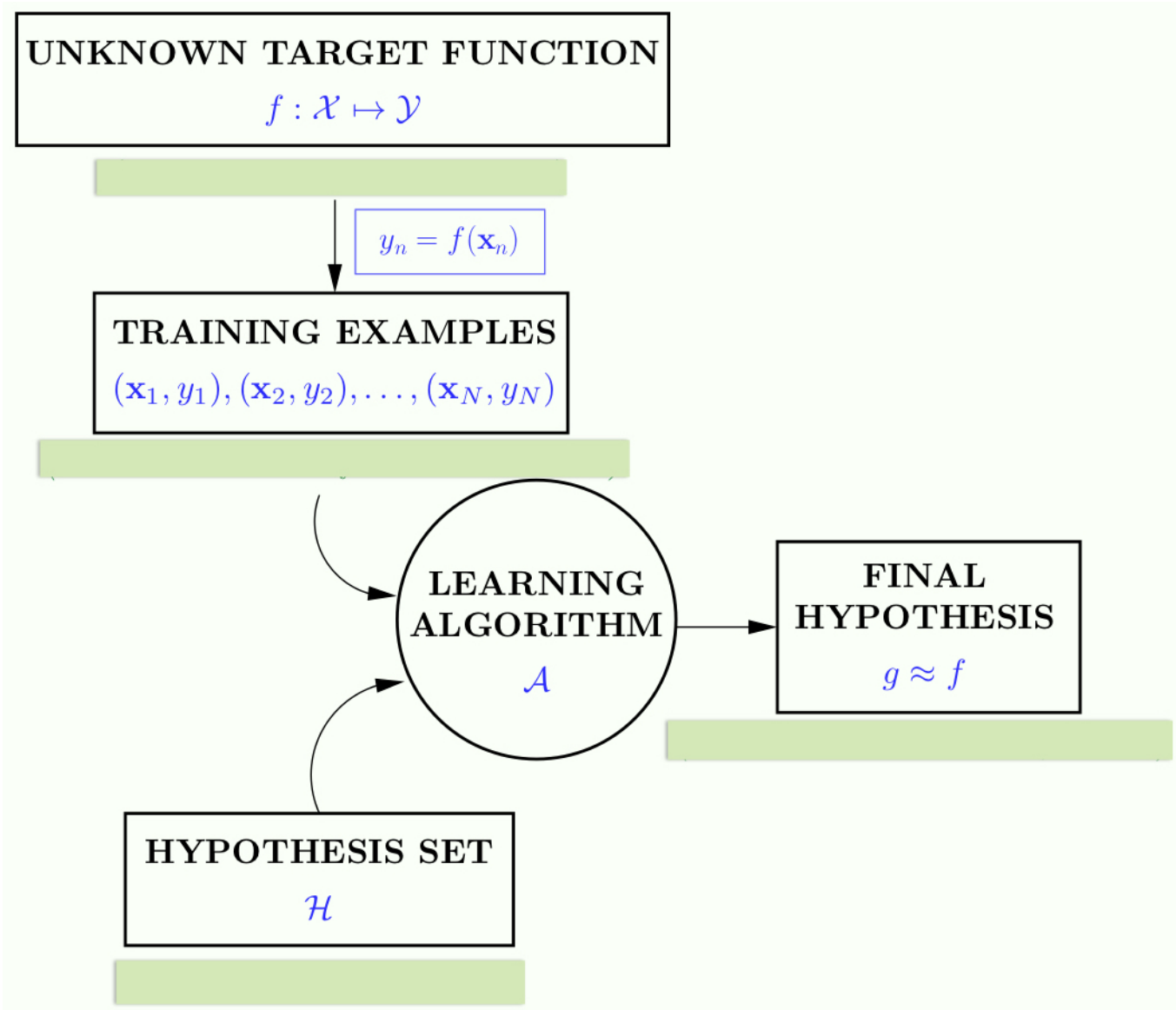
  $$\mathcal{H} = \{h_1, h_2, \ldots\}$$

  is called a **hypothesis set** or a **model**[3].
- Based on $\mathcal{D}$, use an algorithm to select a hypothesis g from $\mathcal{H}$. Goal: $g \approx f$.

[3] Let $\theta \in \Theta$ be a set of parameters. If $h_1, h_2, \ldots$ are functions of θ, such that $h_1 = h(\theta_1), h_2 = h(\theta_2), \ldots$, then we say $\mathcal{H} = \{h_1, h_2, \ldots\}$ is **parametrized** by θ and can be written as $\mathcal{H} = \{h(\theta) : \theta \in \Theta\}$.

© Jiaming Mao

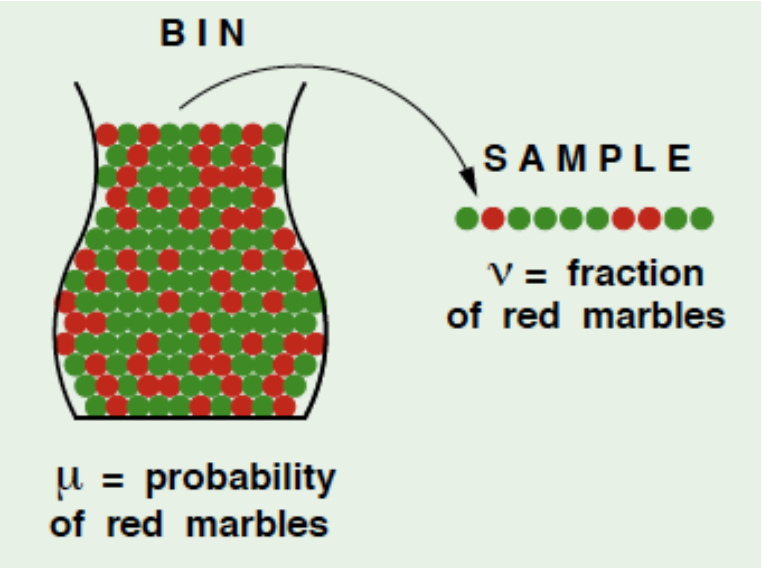
The Learning Problem



© Jiaming Mao

Is Learning Feasible?

- bin with red and green marbles.
- pick a sample of N marbles *independently*.
- μ : probability of picking a red marble
- ν : fraction of red marbles in the sample



- Can we say anything about μ after observing ν?
  - No, sample can be mostly green while bin is mostly red.
  - Yes, sample frequency ν is likely close to bin frequency μ.
  - possible vs. probable

© Jiaming Mao

Probability to the Rescue

Hoeffding's Inequality

Let $z_1, \ldots, z_N$ be N independent Bernoulli random variables with $\Pr(z_i = 1) = \mu$ and $\Pr(z_i = 0) = 1 - \mu$. Let $\nu = \frac{1}{N} \sum_{i=1}^{N} z_i$. Then for any $\epsilon > 0$[a,b],

$$\Pr(|\nu - \mu| > \epsilon) \leq 2e^{-2\epsilon^2 N}$$

---

[a] Note that ν is random, but observed. μ is fixed, but unobserved.

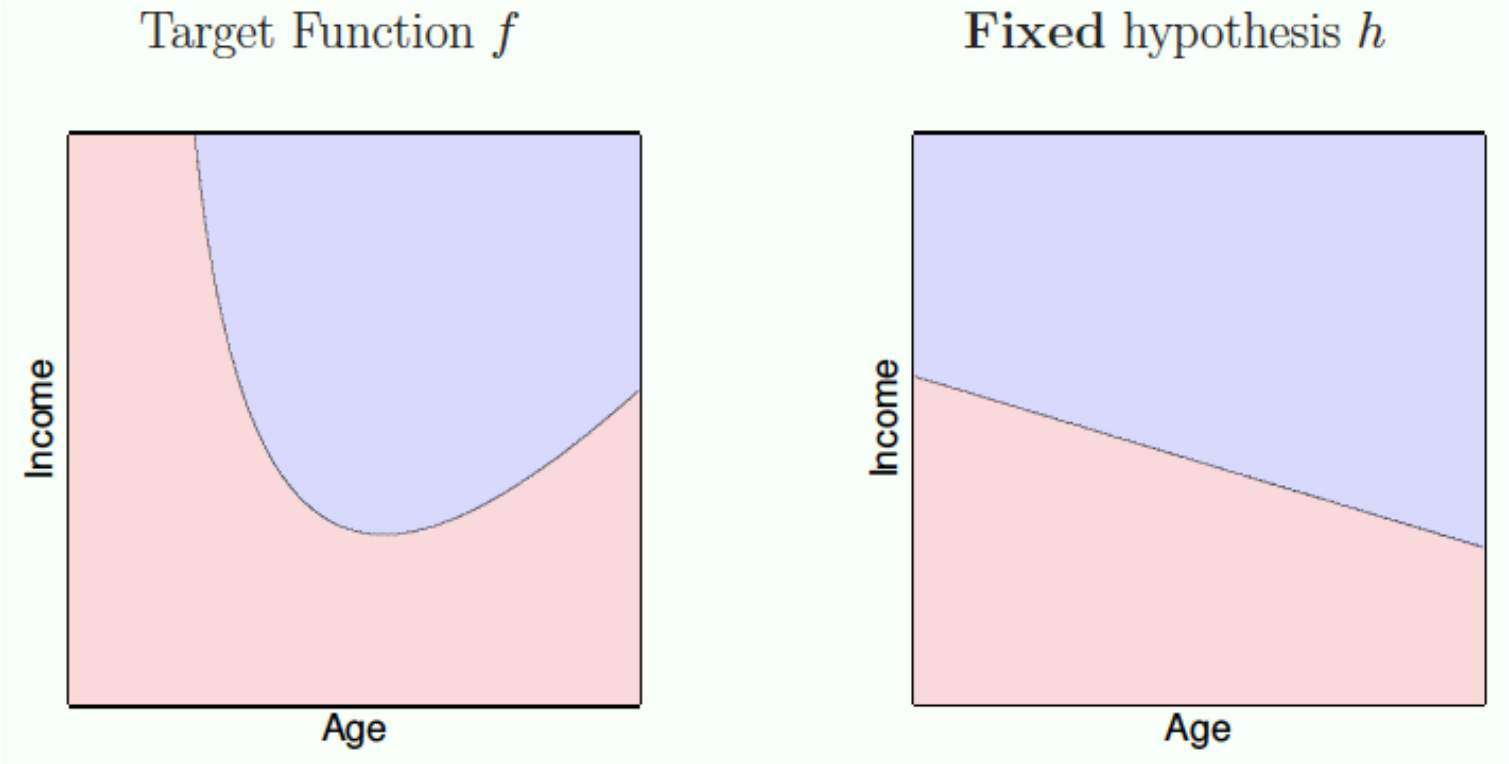
[b] e.g., draw a sample of $N = 1000$ and observe ν. Then,

$$\Pr(|\nu - \mu| > 0.05) \leq 0.014$$
$$\Pr(|\nu - \mu| > 0.10) \leq 0.000000004$$

As we will see, learning is feasible in a probabilistic sense.

© Jiaming Mao

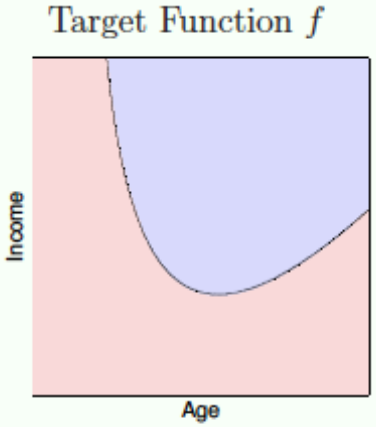
Connection to Learning

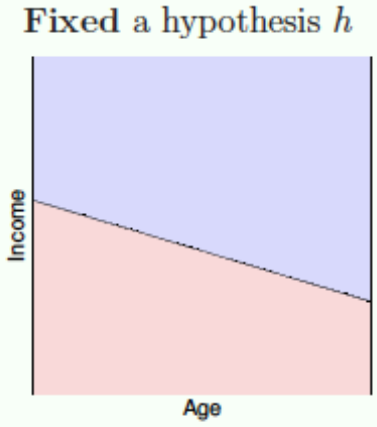


In learning, the unknown is an entire function f

© Jiaming Mao

Connection to Learning





green "marble": $h(\mathbf{x}) = f(\mathbf{x})$

red "marble": $h(\mathbf{x}) \neq f(\mathbf{x})$

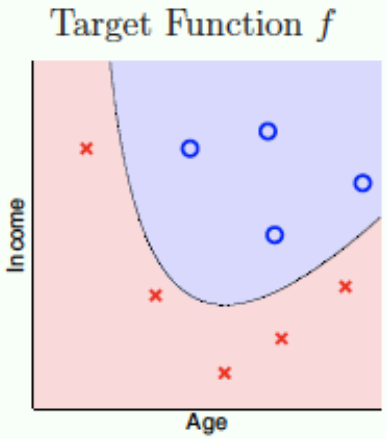
BIN: $\mathcal{X}$

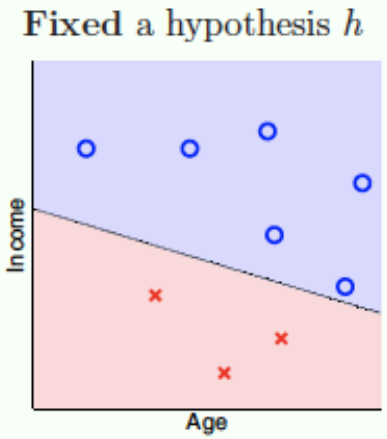
$$E_{\text{out}}(h) = \mathbb{P}_{\mathbf{x}}[h(\mathbf{x}) \neq f(\mathbf{x})]$$

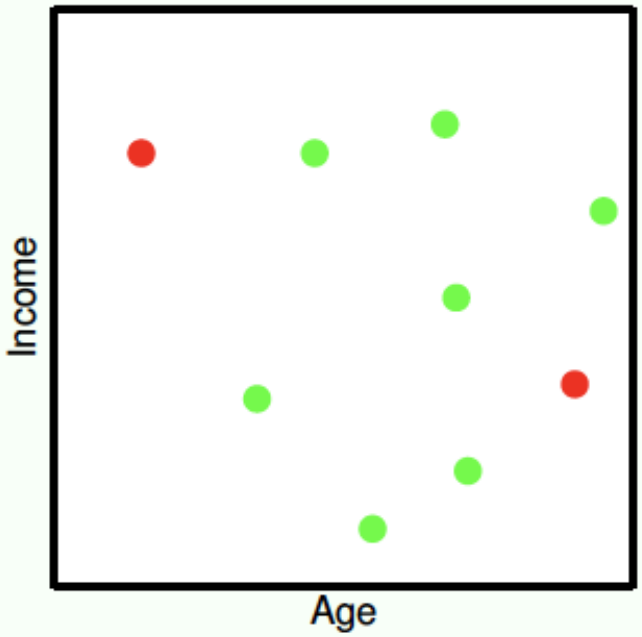
out-of-sample

© Jiaming Mao

Connection to Learning







green data: $h(\mathbf{x}_n) = f(\mathbf{x}_n)$

red data: $h(\mathbf{x}_n) \neq f(\mathbf{x}_n)$

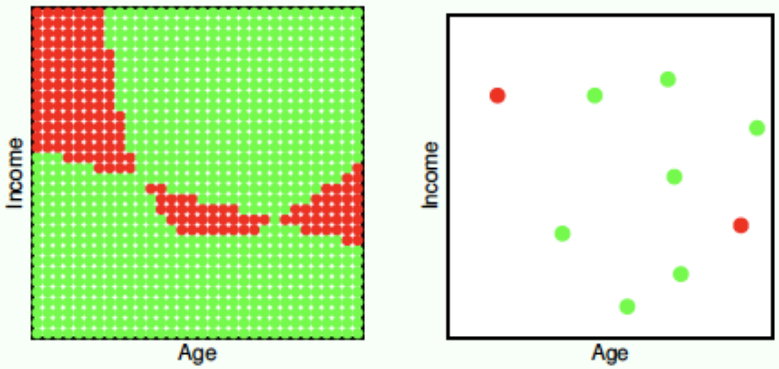
$E_{\text{in}}(h) =$ fraction of red data

in-sample

misclassified

© Jiaming Mao

Connection to Learning



Unknown f and $P(\mathbf{x})$, **fixed** h

Learning

input space $\mathcal{X}$

$\mathbf{x}$ for which $h(\mathbf{x}) = f(\mathbf{x})$

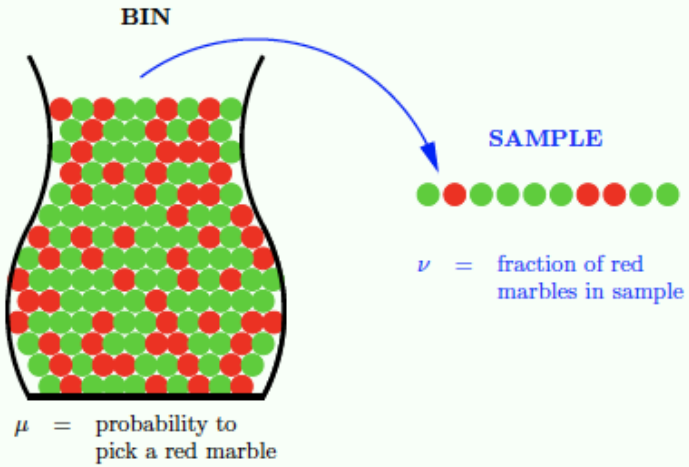
$\mathbf{x}$ for which $h(\mathbf{x}) \neq f(\mathbf{x})$

$P(\mathbf{x})$

data set $\mathcal{D}$

Out-of-sample Error: $E_{\text{out}}(h) = \mathbb{P}_{\mathbf{x}}[h(\mathbf{x}) \neq f(\mathbf{x})]$

In-sample Error: $E_{\text{in}}(h) = \frac{1}{N} \sum_{n=1}^{N} [\![h(\mathbf{x}) \neq f(\mathbf{x})]\!]$



Bin Model

Bin

green marble

red marble

randomly picking a marble

sample of N marbles

μ = probability of picking a red marble

ν = fraction of red marbles in the sample

© Jiaming Mao

Connection to Learning

According to Hoeffding's inequality,

$$\Pr\left(\left|E_{in}\left(h\right) - E_{out}\left(h\right)\right| > \epsilon\right) \leq 2e^{-2\epsilon^2 N} \tag{1}$$

- E_{in} : **in-sample error**; **training error**; **empirical error**; **empirical risk**
- E_{out} : **out-of-sample error**; **expected error**; **prediction error**; **risk**

(1) says that for a given h, given large enough N,

- $E_{in} \approx E_{out}$.
- If $E_{in} \approx 0$, then $E_{out} \approx 0$. In this case we have learned something about f: $f \approx h$ over $\mathcal{X}$.

© Jiaming Mao

Connection to Learning

The key assumptions that are needed for (1) to hold are:

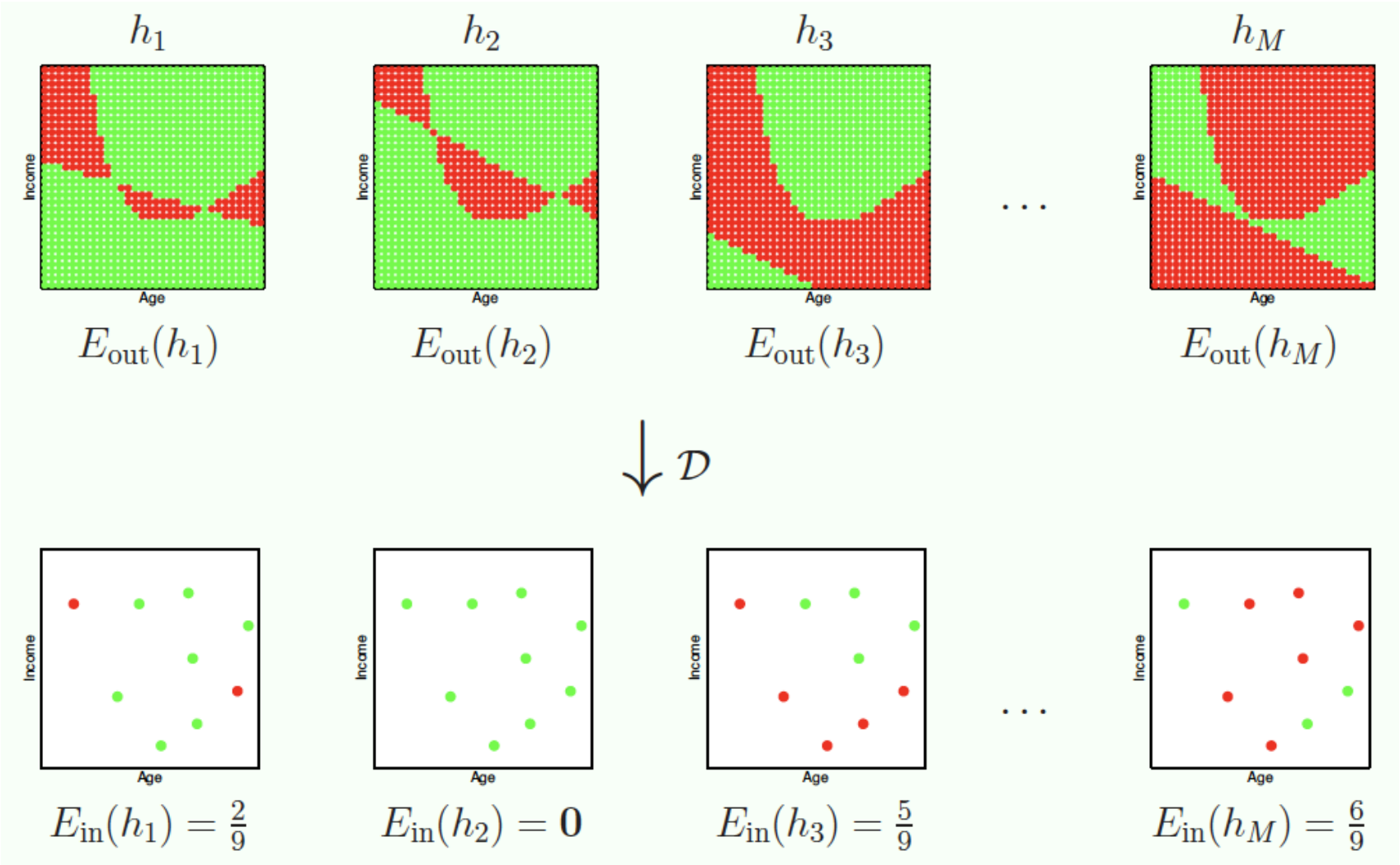
1. The data set $\mathcal{D}$ is a random sample, i.e. data points are drawn $i.i.d.$ from the underlying distribution.
   - In order to say something about unobserved data, we need to *assume* that they resemble the observed data.
   - Assumptions on the data generating process (here: drawn $i.i.d.$) is always necessary if we want to say anything beyond our observed data.
2. h is fixed (before $\mathcal{D}$ is generated).

If the assumptions are satisfied, then (1) says that a sample $\mathcal{D}$ can be used to assess whether or not h is close to f.

- However, this is verification, not learning.

© Jiaming Mao

Finite Learning Model



If we pick the hypothesis with minimum E_{in}, will E_{out} be small?

© Jiaming Mao

Finite Learning Model

- If you toss a fair coin 10 times, what is the probability that you will get 10 heads?
  - $\approx 0.1\%$
- If you toss 1000 fair coins 10 times each, what is the probability that *some* coin will get 10 heads?
  - $\approx 62\%$

© Jiaming Mao

Finite Learning Model

Let $g \in \mathcal{H} = \{h_1, \ldots, h_M\}$.

$$\begin{aligned}
\Pr\left(|E_{in}(g) - E_{out}(g)| > \epsilon\right) \leq \Pr\{&|E_{in}(h_1) - E_{out}(h_1)| > \epsilon \\
&\text{or } |E_{in}(h_2) - E_{out}(h_2)| > \epsilon \\
&\ldots \\
&\text{or } |E_{in}(h_M) - E_{out}(h_M)| > \epsilon\} \\
\leq \sum_{m=1}^{M} &\Pr\left(|E_{in}(h_m) - E_{out}(h_m)| > \epsilon\right) \\
\leq 2|\mathcal{H}|&e^{-2\epsilon^2 N} \qquad (2)
\end{aligned}$$

, where $|\mathcal{H}| = M$ is the size of $\mathcal{H}$.

- (2) is valid for any $g \in \mathcal{H}$, regardless how g is selected.
- Note g is *not* fixed before the data is generated: the selection of g depends on $\mathcal{D}$.

© Jiaming Mao

Finite Learning Model

Let $\delta \equiv 2\left|\mathcal{H}\right| e^{-2\epsilon^2 N}$. (2) $\Rightarrow$ with probability at least $1-\delta$,

$$E_{out}\left(g\right) \leq E_{in}\left(g\right) + \sqrt{\frac{1}{2N} \ln \frac{2\left|\mathcal{H}\right|}{\delta}} \tag{3}$$

- (3) is referred to as a **generalization bound**.

© Jiaming Mao

The Learning Problem

The feasibility of learning is split into two questions:

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

- $|\mathcal{H}|$ can be thought of as a measure of the **complexity** of $\mathcal{H}$.
- Tradeoff:
  - Small $|\mathcal{H}| \Rightarrow E_{in}(g) \approx E_{out}(g)$
  - Large $|\mathcal{H}| \Rightarrow$ more likely to find g such that $E_{in}(g) \approx 0$

© Jiaming Mao

Effective Number of Hypotheses

- In practice, hypothesis sets are typically infinite in size.
- How to derive the generalization bound when $\mathcal{H}$ is infinite?

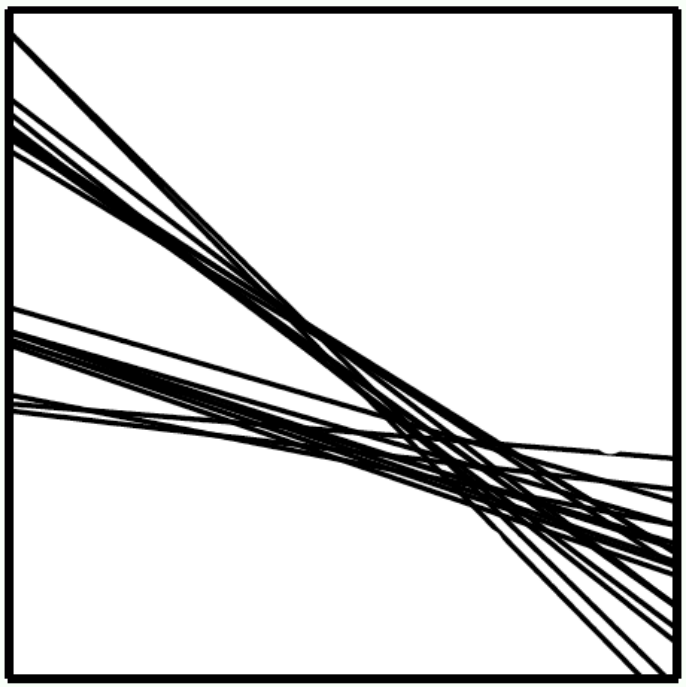
  Idea:

- On a given data set $\mathcal{D}$, many $h \in \mathcal{H}$ will look the same, i.e., they map $\mathcal{D}$ into the same set of values.
- These hypotheses are identical from the data's perspective. Therefore there are "effectively" fewer than $|\mathcal{H}|$ hypotheses[4].

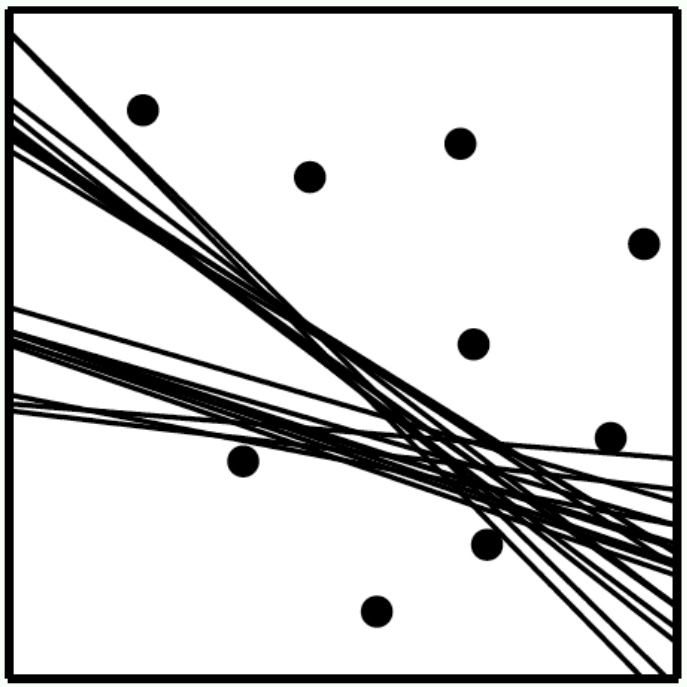
[4] Since the data is all we have.

© Jiaming Mao

Effective Number of Hypotheses



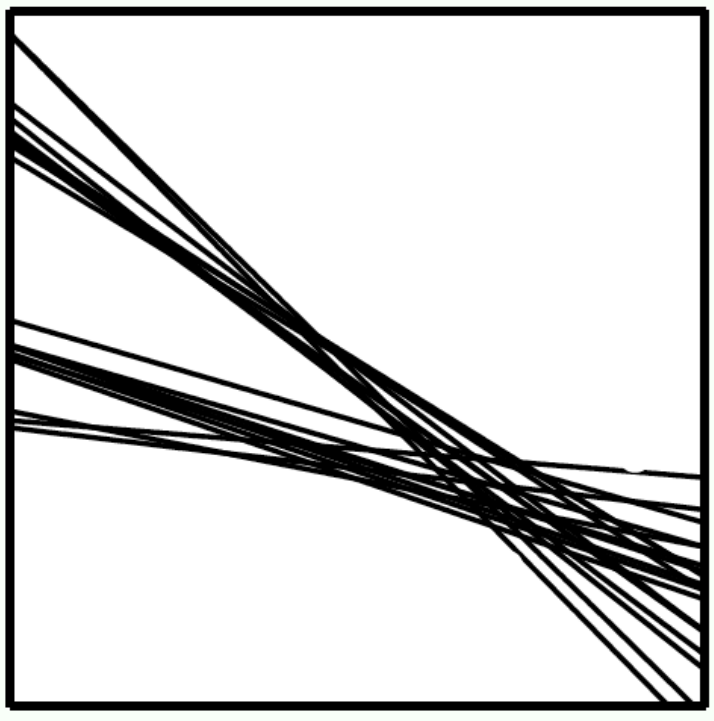
$\mathcal{H}$

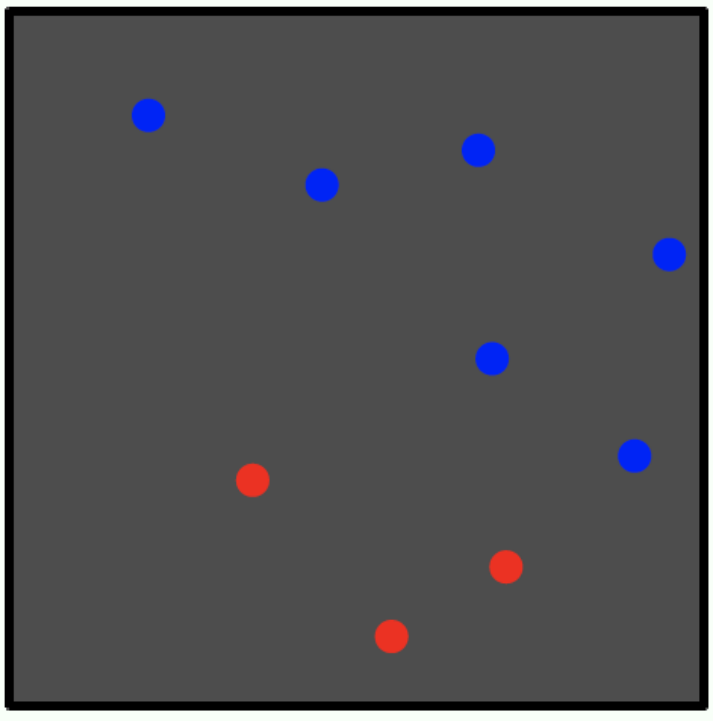


$\mathcal{H}$ through the eyes of the $\mathcal{D}$

© Jiaming Mao

Effective Number of Hypotheses





From the point of view of $\mathcal{D}$, the entire $\mathcal{H}$ is just one dichotomy.

© Jiaming Mao

Growth Function

- Consider binary target functions and hypothesis sets that contain $h : \mathcal{X} \to \{-1, +1\}$.
- If $h \in \mathcal{H}$ is applied to a finite sample $x_1, \ldots, x_N \in \mathcal{X}$, we get an $N-$tuple $h(x_1), \ldots, h(x_N)$ of ± 1's.
- Such an $N-$tuple is called a dichotomy since it splits $x_1, \ldots, x_N$ into two groups: those points for which h is -1 and those for which h is $+1$.
- Each $h \in \mathcal{H}$ generates a dichotomy on $x_1, \ldots, x_N$, but two different h's may generate the same dichotomy if they happen to give the same pattern of ± 1's on this particular sample.

© Jiaming Mao

VC Dimension

- The **growth function** for a hypothesis set $\mathcal{H}$, denoted $m_{\mathcal{H}}(N)$, is the maximum possible number of dichotomies $\mathcal{H}$ can generate on a data set of N points[5].
- If $\mathcal{H}$ is capable of generating all possible dichotomies on $x_1, \ldots, x_N$, then $\mathcal{H}$ **shatters** $x_1, \ldots, x_N$, in which case $m_{\mathcal{H}}(N) = 2^N$.
- The **Vapnik-Chervonenkis (VC) dimension** of $\mathcal{H}$, denoted $d_{vc}(\mathcal{H})$, is the size of the largest data set that $\mathcal{H}$ can shatter[6].
  - $d_{vc}(\mathcal{H})$ is the largest value of N for which $m_{\mathcal{H}}(N) = 2^N$.

[5] rather than over the entire input space $\mathcal{X}$.

[6] See Appendix I for a detailed introduction to growth function and VC dimension.

© Jiaming Mao

VC Inequality

The Vapnik-Chervonenkis Inequality

Let $\mathcal{H}$ be a set of binary-valued hypotheses. For any $g \in \mathcal{H}$,

$$\Pr\left(\left|E_{in}\left(g\right) - E_{out}\left(g\right)\right| > \epsilon\right) \leq 4 m_{\mathcal{H}}\left(2N\right) e^{-\frac{1}{8}\epsilon^2 N} \tag{4}$$

© Jiaming Mao

VC Bound

VC Generalization Bound

(4) $\Rightarrow$ for any tolerance $\delta > 0$,

$$E_{out}(g) \leq E_{in}(g) + \sqrt{\frac{8}{N} \ln \frac{4 m_{\mathcal{H}}(2N)}{\delta}} \tag{5}$$

with probability $\geq 1 - \delta$.

© Jiaming Mao

VC Bound

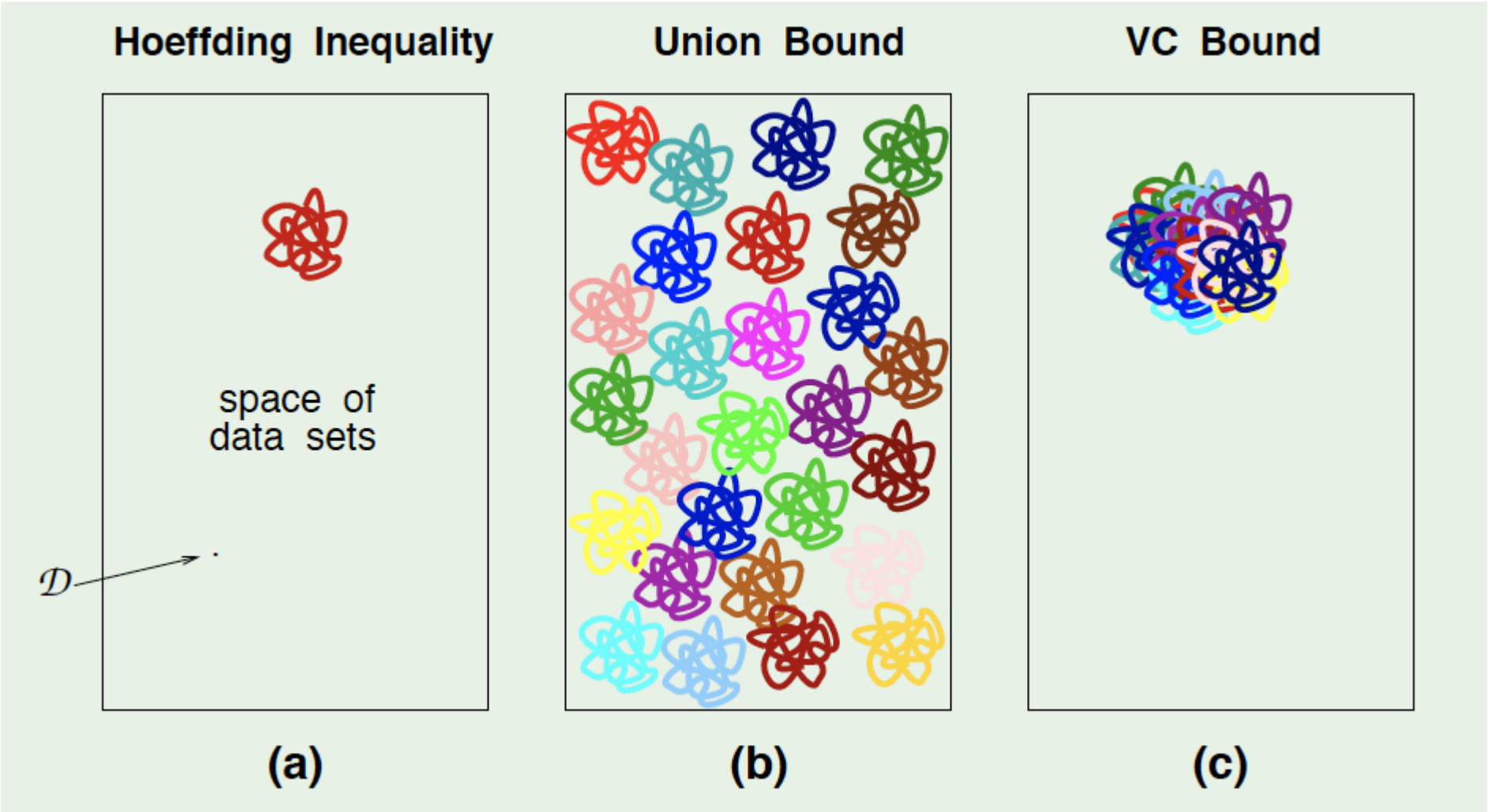
- We can prove that:

$$m_{\mathcal{H}}(N) \leq \begin{cases} N^{d_{vc}(\mathcal{H})} + 1 \\ \left(\frac{eN}{d_{vc}(\mathcal{H})}\right)^{d_{vc}(\mathcal{H})} \end{cases} \quad N \geq d_{vc}(\mathcal{H}) \tag{6}$$

- The VC inequality and VC generalization bound establish the feasibility of learning with infinite hypothesis sets: with enough data, each and every hypothesis in an infinite $\mathcal{H}$ with a finite VC dimension will generate well from E_{in} to E_{out}.

© Jiaming Mao

VC Bound



© Jiaming Mao

Training versus Testing

If we have an independent test set not used for selecting g from $\mathcal{H}$, and on which we can evaluate the performance of g, then

$$\text{Training:} \quad \Pr\left(\left|E_{in}\left(g\right) - E_{out}\left(g\right)\right| > \epsilon\right) \leq 4 m_{\mathcal{H}}\left(2N\right) e^{-\frac{1}{8}\epsilon^2 N}$$

$$\text{Testing:} \quad \Pr\left(\left|E_{test}\left(g\right) - E_{out}\left(g\right)\right| > \epsilon\right) \leq 2 e^{-2\epsilon^2 N}$$

- The generalization bound for test error is much tighter.
- The test set is not biased, whereas the training set has an *optimistic* bias, since it is used to choose a hypothesis that looks good *on it*.
- The price for a test set is fewer data for training.

© Jiaming Mao

Approximation-Generalization Tradeoff

(5) and (6) $\Rightarrow$

$$E_{out}(g) \leq E_{in}(g) + \sqrt{\frac{8}{N} \ln \frac{4 m_{\mathcal{H}}(2N)}{\delta}}$$

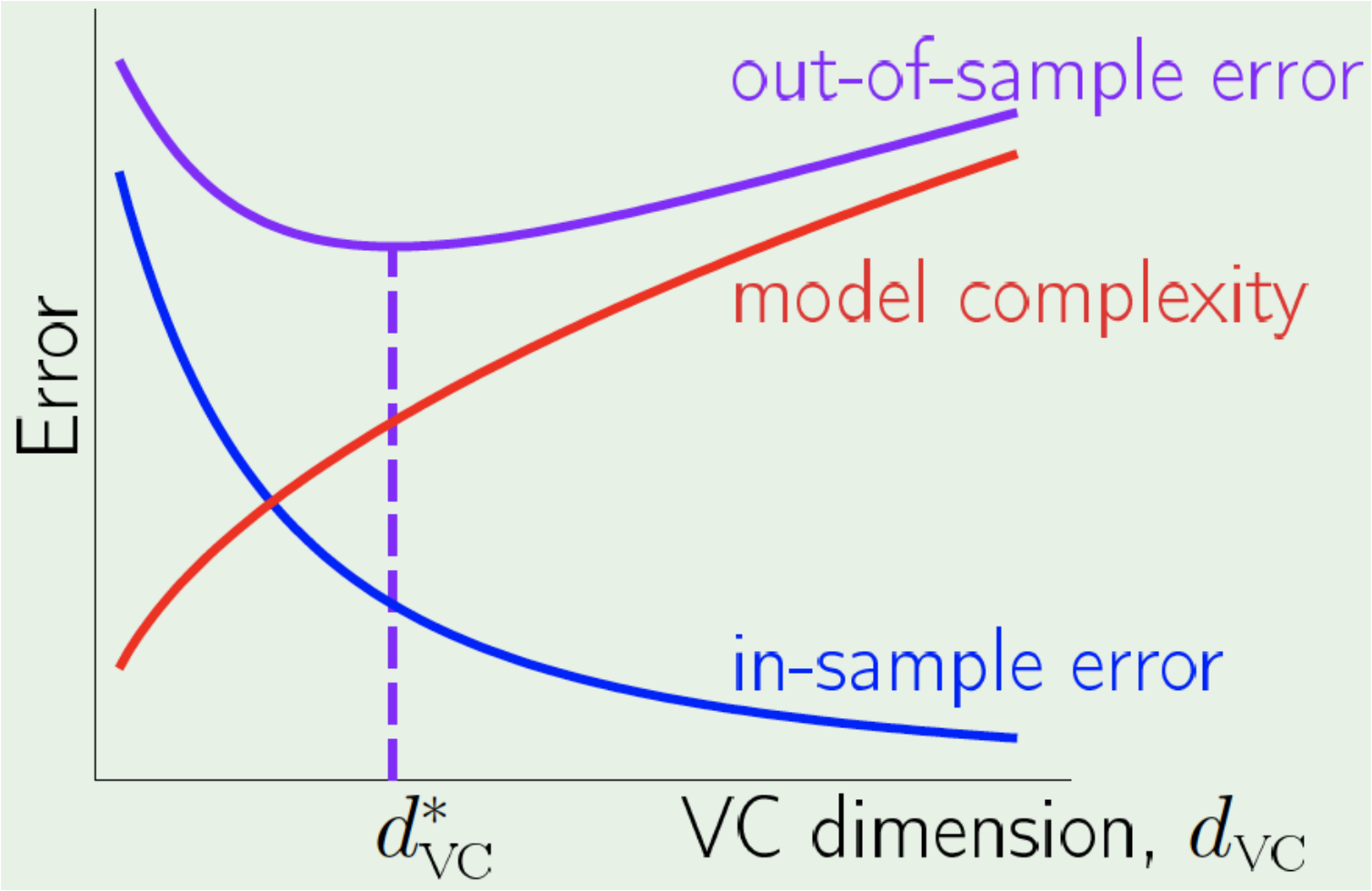
$$\leq E_{in}(g) + \underbrace{\sqrt{\frac{8}{N} \ln \frac{4\left((2N)^{d_{vc}} + 1\right)}{\delta}}}_{\Omega(d_{vc})}$$

$\Omega(d_{vc})$ can be viewed as a penalty for model complexity.

- More complex $\mathcal{H}$ ($d_{vc} \uparrow$) $\Rightarrow$ better chance of approximating f in sample ($E_{in} \approx 0$)
- Less complex $\mathcal{H}$ ($d_{vc} \downarrow$) $\Rightarrow$ better chance of generalizing out of sample ($E_{in} \approx E_{out}$)

© Jiaming Mao

Approximation-Generalization Tradeoff



© Jiaming Mao

Approximation-Generalization Tradeoff

- VC analysis shows the choice of $\mathcal{H}$ needs to strike a balance between approximating f on the training data and generalizing to new data.
- If $\mathcal{H}$ is too simple, we may fail to approximate f well on the training data and end up with a large in-sample error term.
- If $\mathcal{H}$ is too complex, we may fail to generalize well because of the large model complexity term.

© Jiaming Mao

Learning as Optimization

The choice of error measure that quantifies how well a hypothesis h approximates the target function f matters for the learning process and can affect the final hypothesis g that is chosen.

Formally,

$$E_{out}(h) = \mathbb{E}\left[\ell\left(h\left(x\right), f\left(x\right)\right)\right]$$

$$E_{in}(h) = \frac{1}{N}\sum_{i=1}^{N}\ell\left(h\left(x_i\right), f\left(x_i\right)\right)$$

, where $\ell\left(h\left(x\right), f\left(x\right)\right)$ is a **loss function** that measures the difference between $h\left(x\right)$ and $f\left(x\right)$[7].

[7] Hence E_{out} = expected loss. E_{in} =average loss on the training data.

© Jiaming Mao

Learning as Optimization

- Some commonly used loss functions are:[8]

$$\ell(x, y) = (x - y)^2 \quad \textbf{squared-error loss}$$
$$\ell(x, y) = |x - y| \quad \textbf{absolute-error loss}$$
$$\ell(x, y) = \mathcal{I}(x \neq y) \quad \textbf{zero-one loss}$$

- We have used zero-one loss in VC analysis when dealing with binary target functions. For real-valued functions, a common choice is to use the squared-error loss.

- The process of learning is also a process of optimization: we choose g by minimizing an objective function, which is the error measure based on our chosen loss function.

[8] The squared-error loss is also called **quadratic loss** or **L2 loss**. The absolute-error loss is also called **linear loss** or **L1 loss**.

© Jiaming Mao

Bias-Variance Decomposition

Bias-variance decomposition provides another way of looking at the approximation-generalization tradeoff.

Consider a real-valued target function f. Let $g \in \mathcal{H}$ be the hypothesis chosen to approximate f. Then

$$\begin{aligned} E_{out}(g) &= \mathbb{E}\left[(g(x) - f(x))^2\right] \\ &= \mathbb{V}(g(x)) + \mathbb{E}[(g(x) - f(x))]^2 \\ &= \mathbb{V}(g) + [\text{bias}(g)]^2 \end{aligned} \tag{7}$$

, where $\text{bias}(g) \doteq \mathbb{E}[(g(x) - f(x))]$.

© Jiaming Mao

Bias-Variance Decomposition

9,10

---

[9] Note: the expectation is with respect to both x and $\mathcal{D}$, since g depends on $\mathcal{D}$. I.e.,

$$\begin{aligned}\text{bias}(g) &= \mathbb{E}_x\left[\mathbb{E}_{\mathcal{D}}\left[(g(x) - f(x))\right]\right] = \mathbb{E}_x\left[\mathbb{E}_{\mathcal{D}}\left[g(x)\right] - f(x)\right] \\ &= \mathbb{E}_x\left[\bar{g}(x) - f(x)\right]\end{aligned}$$

, where $\bar{g}(x) \doteq \mathbb{E}_{\mathcal{D}}\left[g(x)\right]$. Similarly,

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

[10] If

$$y = f(x) + e$$

, where $\mathbb{E}[e] = 0$, then

$$\begin{aligned}E_{out}(g) &= \mathbb{E}\left[\left(y - g(x)\right)^2\right] \\ &= \mathbb{V}(g) + \left[\text{bias}(g)\right]^2 + \mathbb{V}(e)\end{aligned}$$

© Jiaming Mao

Bias-Variance Decomposition

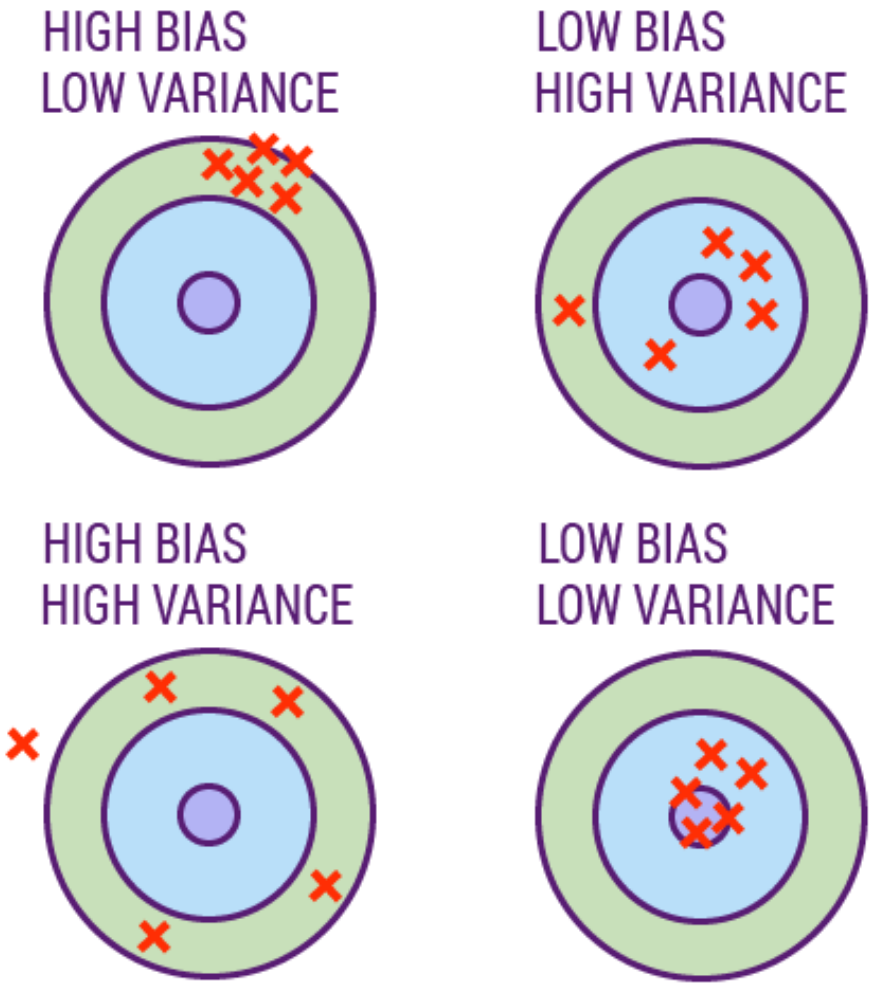
- Intuitively, bias arises if the model $\mathcal{H}$ does not contain f[11]. Thus there will be error even if we fit the model on the entire population.

- $\mathbb{V}(g)$ refers to the amount by which g would change if we estimate it using a different data set. The variance term arises because we have limited data. The g that we select based on a limited sample is almost never the same as the g that we would select if we have access to the entire population.
  - In general, the variance term decreases as sample size increases.

[11] which is almost always the case: our model hardly ever contains the true f.

© Jiaming Mao

Bias-Variance Decomposition



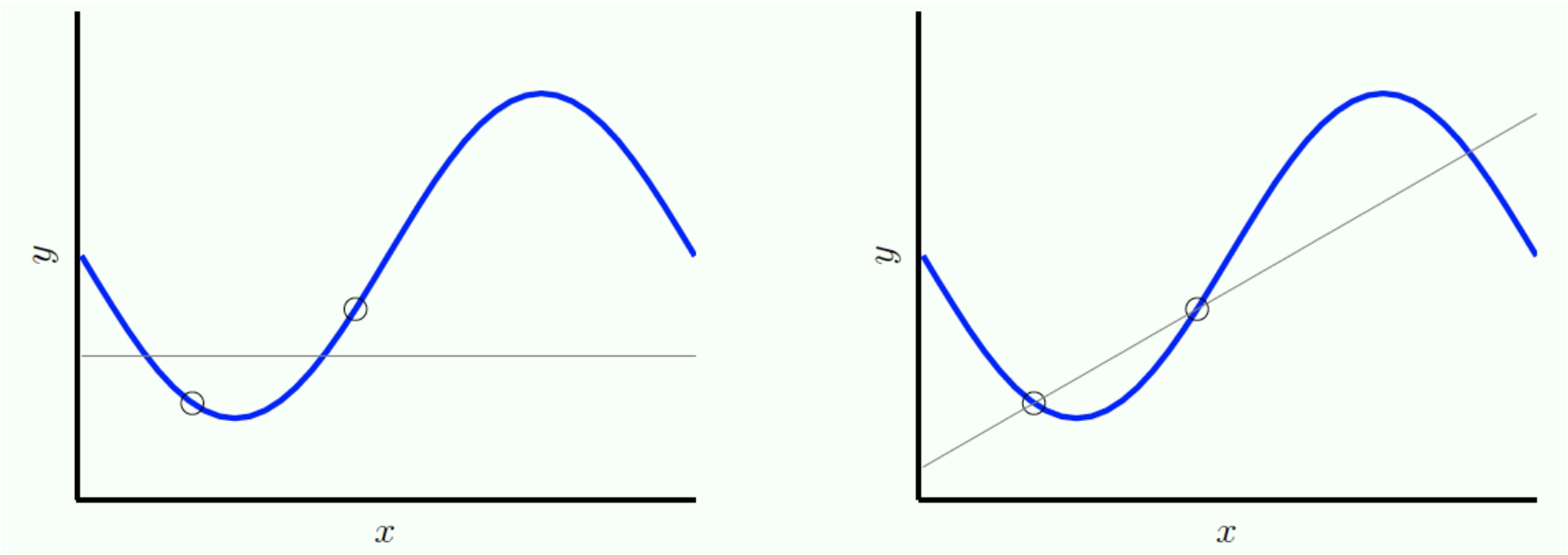
© Jiaming Mao

Bias-Variance Decomposition

$$y = f(x) = \sin(\pi x)$$

- Two models:

$$\mathcal{H}_0 : h(x) = b$$
$$\mathcal{H}_1 : h(x) = ax + b$$

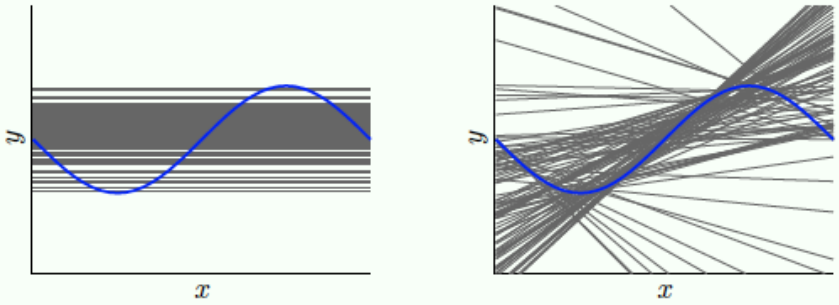


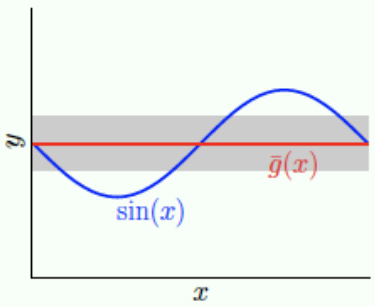
2 data points

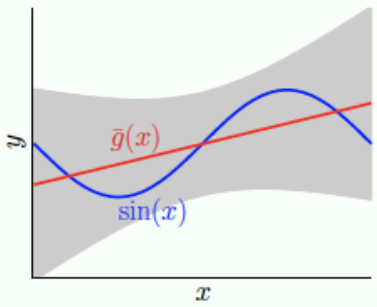
© Jiaming Mao

Bias-Variance Decomposition

2 Data Points







$\mathcal{H}_0$

bias $= 0.50$;

var $= 0.25$.

$E_{\text{out}} = 0.75$ ✓

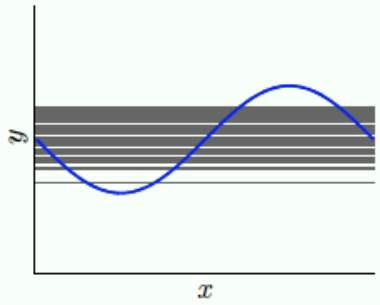
$\mathcal{H}_1$

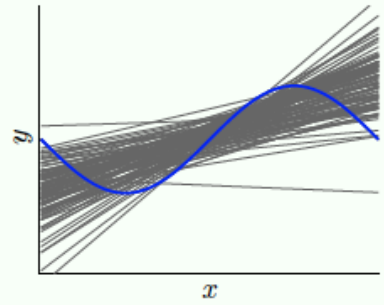
bias $= 0.21$;

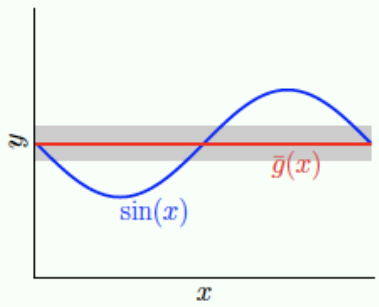
var $= 1.69$.

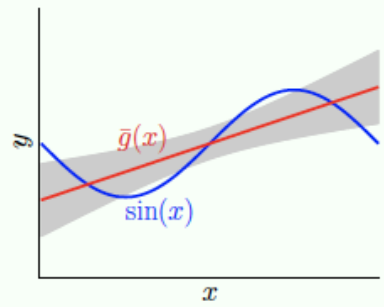
$E_{\text{out}} = 1.90$

5 Data Points









$\mathcal{H}_0$

bias $= 0.50$;

var $= 0.1$.

$E_{\text{out}} = 0.6$

$\mathcal{H}_1$

bias $= 0.21$;

var $= 0.21$.

$E_{\text{out}} = 0.42$ ✓

© Jiaming Mao

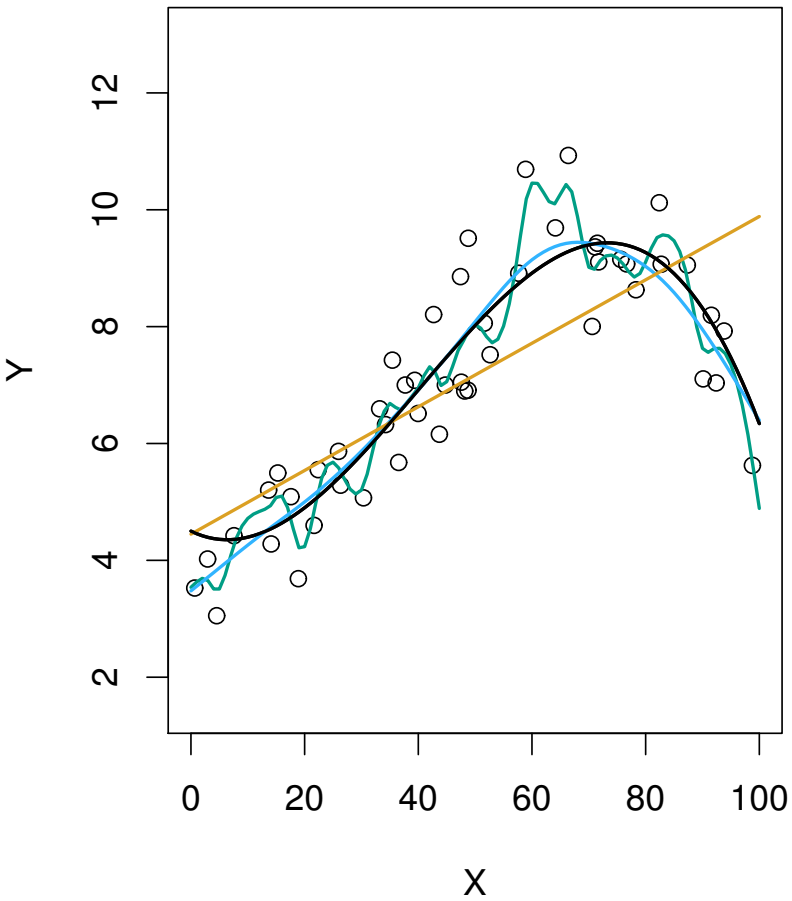
Bias-Variance Trade-off

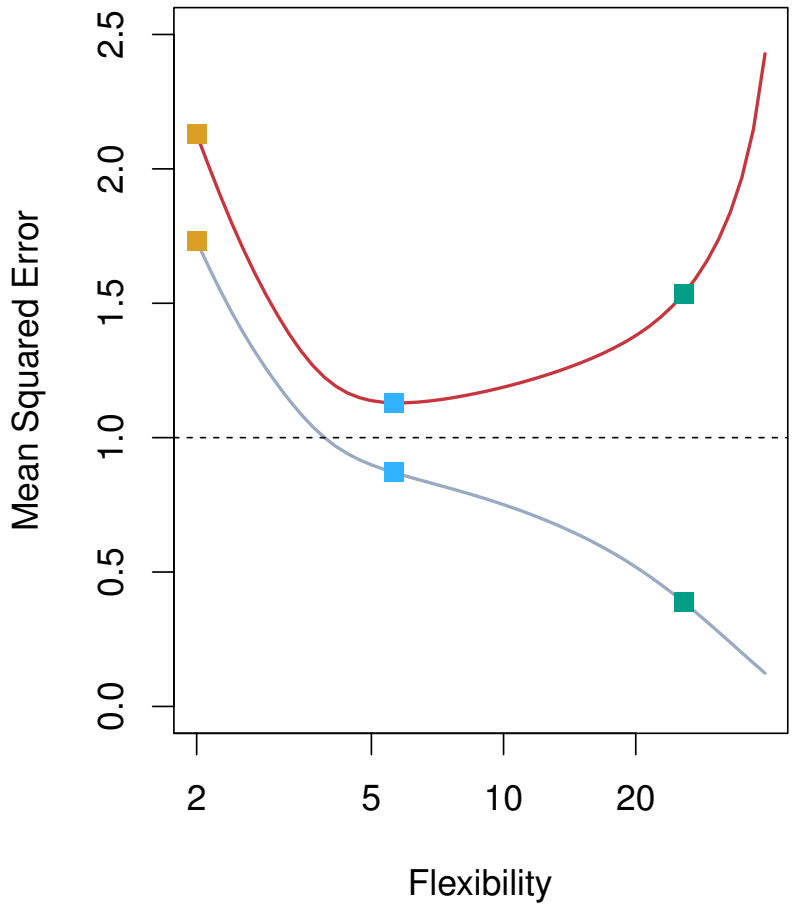
- In general, as model complexity increases, the bias will decrease and the variance will increase, leading to the **bias-variance trade-off**.
  - More complex models tend to have higher variance because they have the capacity to follow the data more closely. Thus using a different set of data points may cause g to change considerably.
  - The challenge lies in finding a model for which both the bias and the are low.

© Jiaming Mao

Bias-Variance Trade-off

$$y = f(x) + e$$



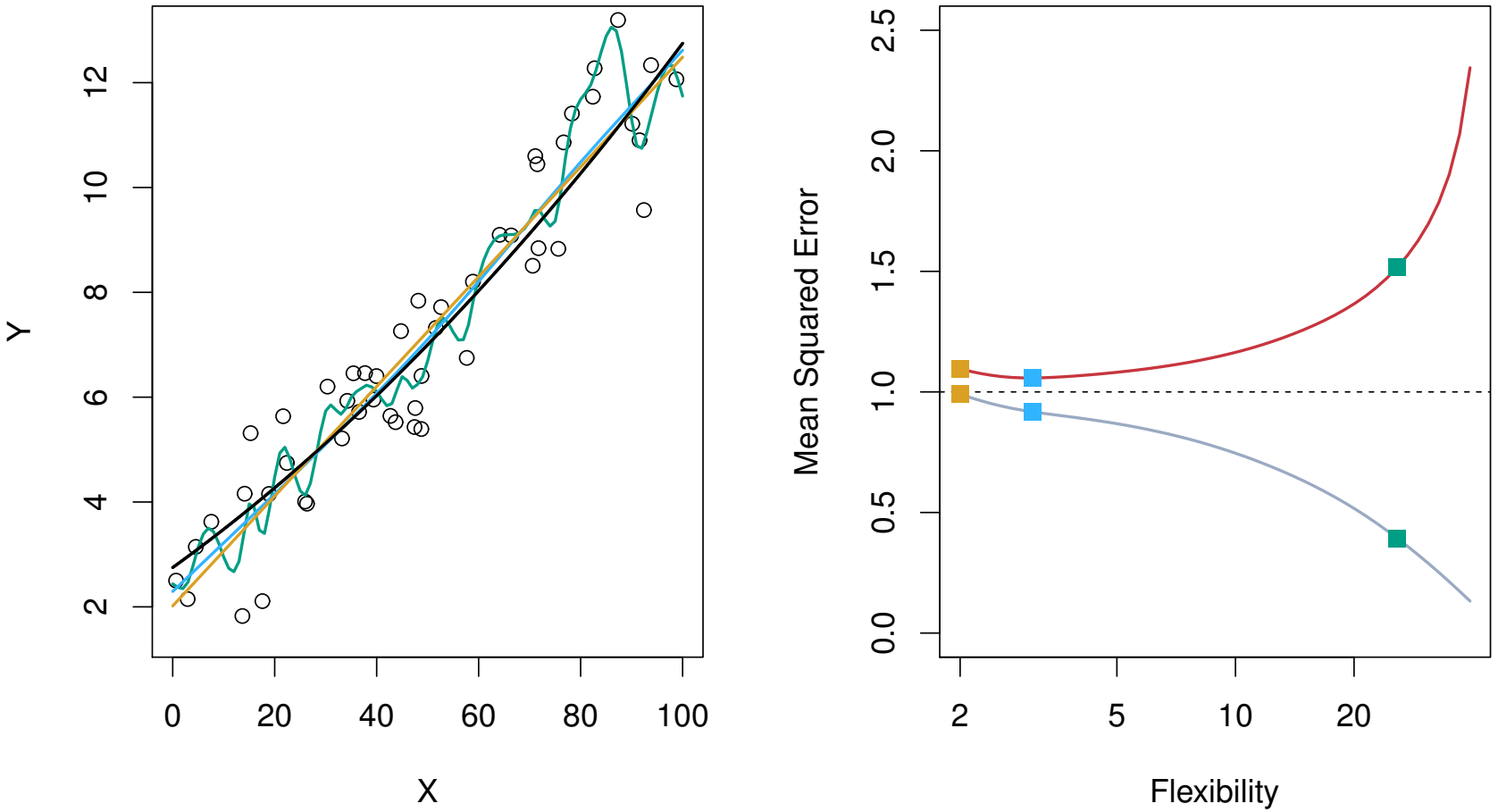


Left: f (black), linear fit (orange), smoothing spline fits (blue & green)
Right: E_{in} (grey), E_{out} (red), $Var(e)$ (dashed)

© Jiaming Mao

Bias-Variance Trade-off

$$y = f(x) + e$$

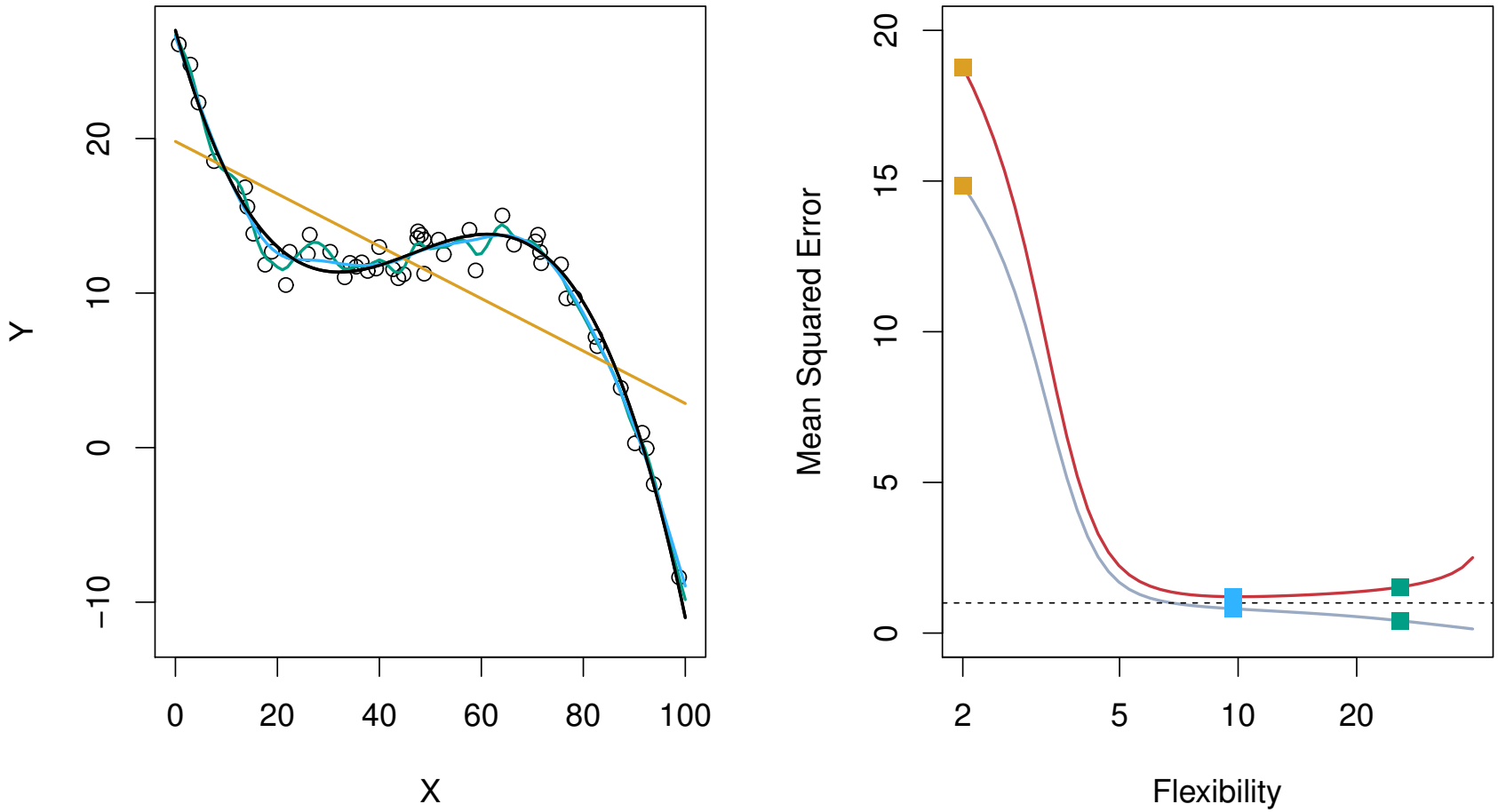


Left: f (black), linear fit (orange), smoothing spline fits (blue & green)
Right: E_{in} (grey), E_{out} (red), $Var(e)$ (dashed)

© Jiaming Mao

Bias-Variance Trade-off

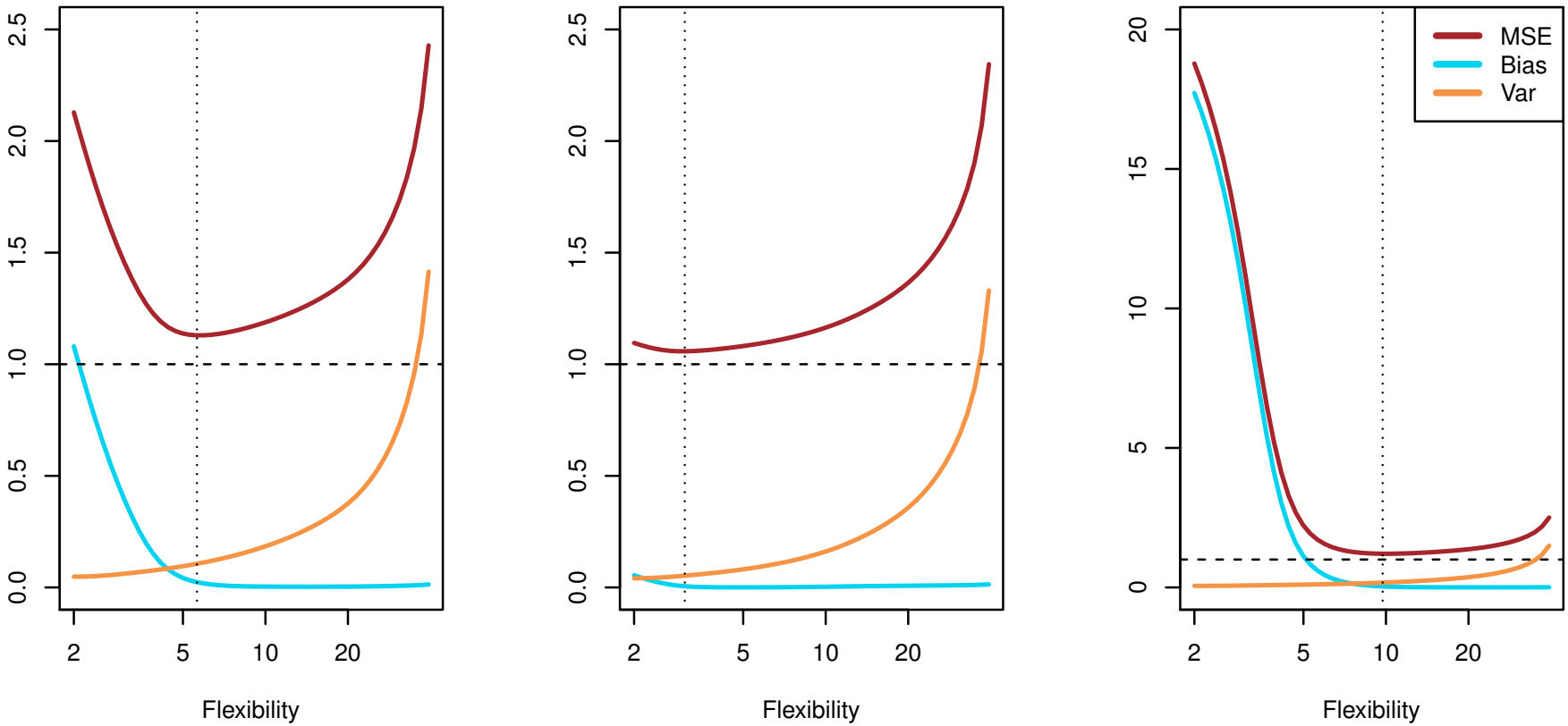
$$y = f(x) + e$$



Left: f (black), linear fit (orange), smoothing spline fits (blue & green)
Right: E_{in} (grey), E_{out} (red), $Var(e)$ (dashed)

© Jiaming Mao

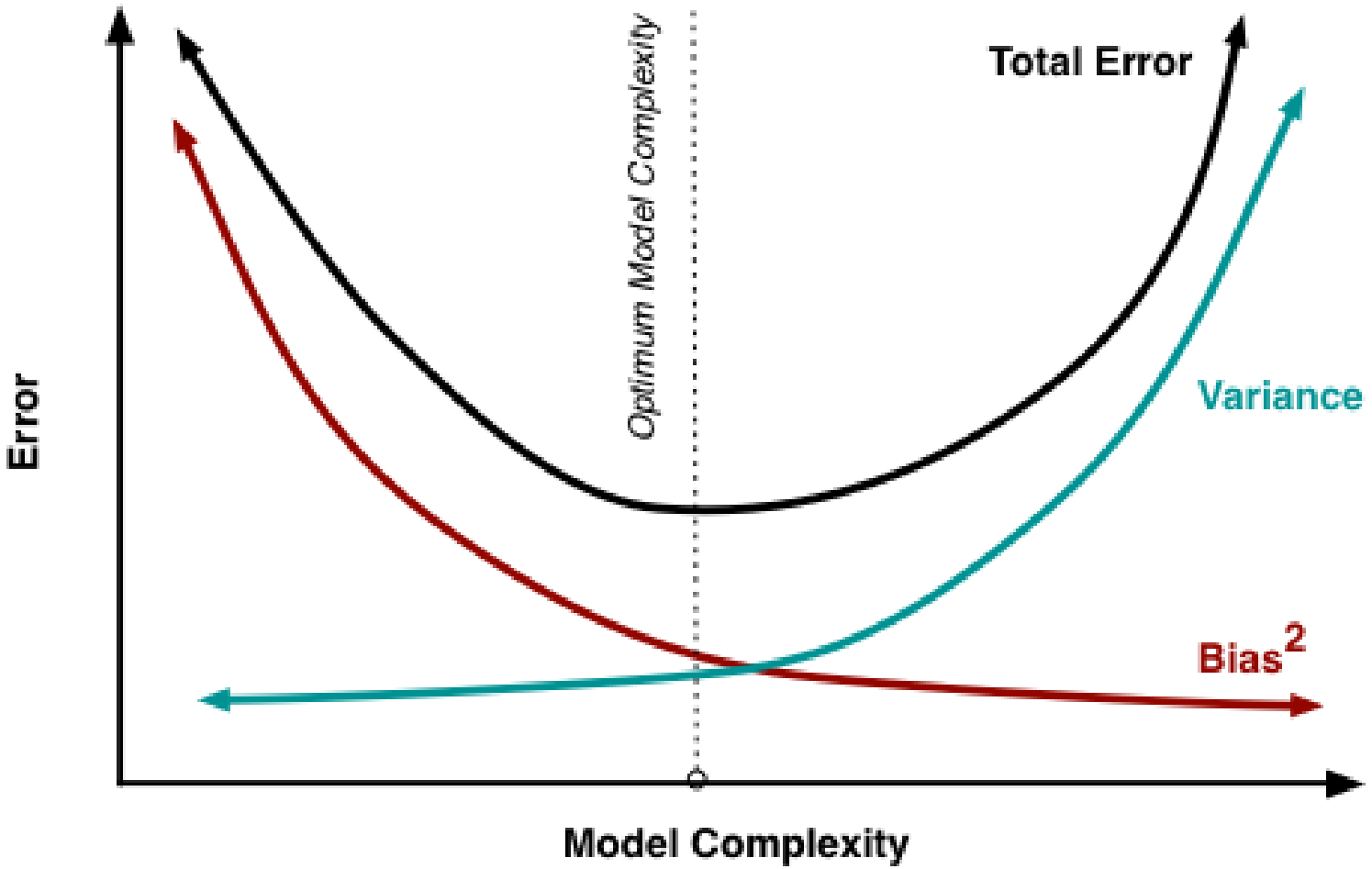
Bias-Variance Trade-off



Bias-variance decomposition for the three examples

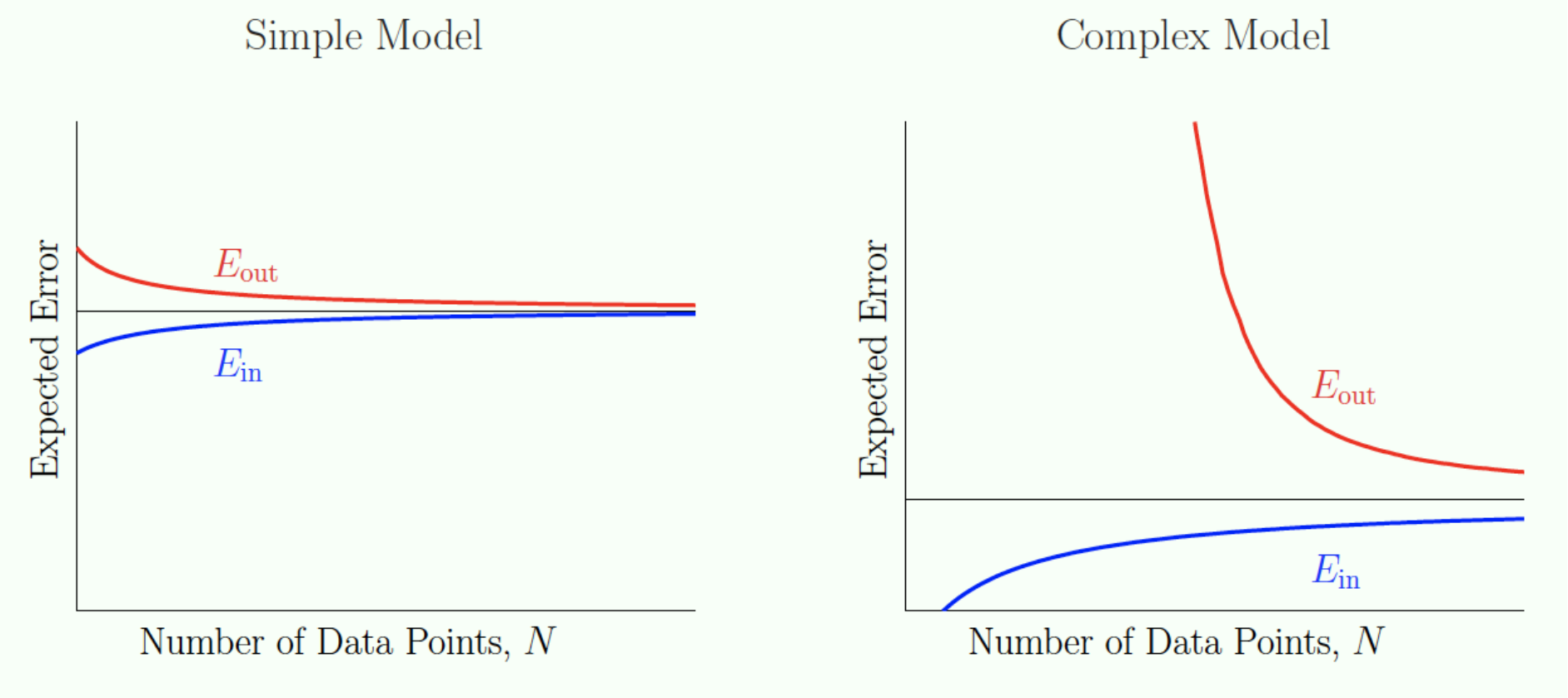
© Jiaming Mao

Bias-Variance Trade-off



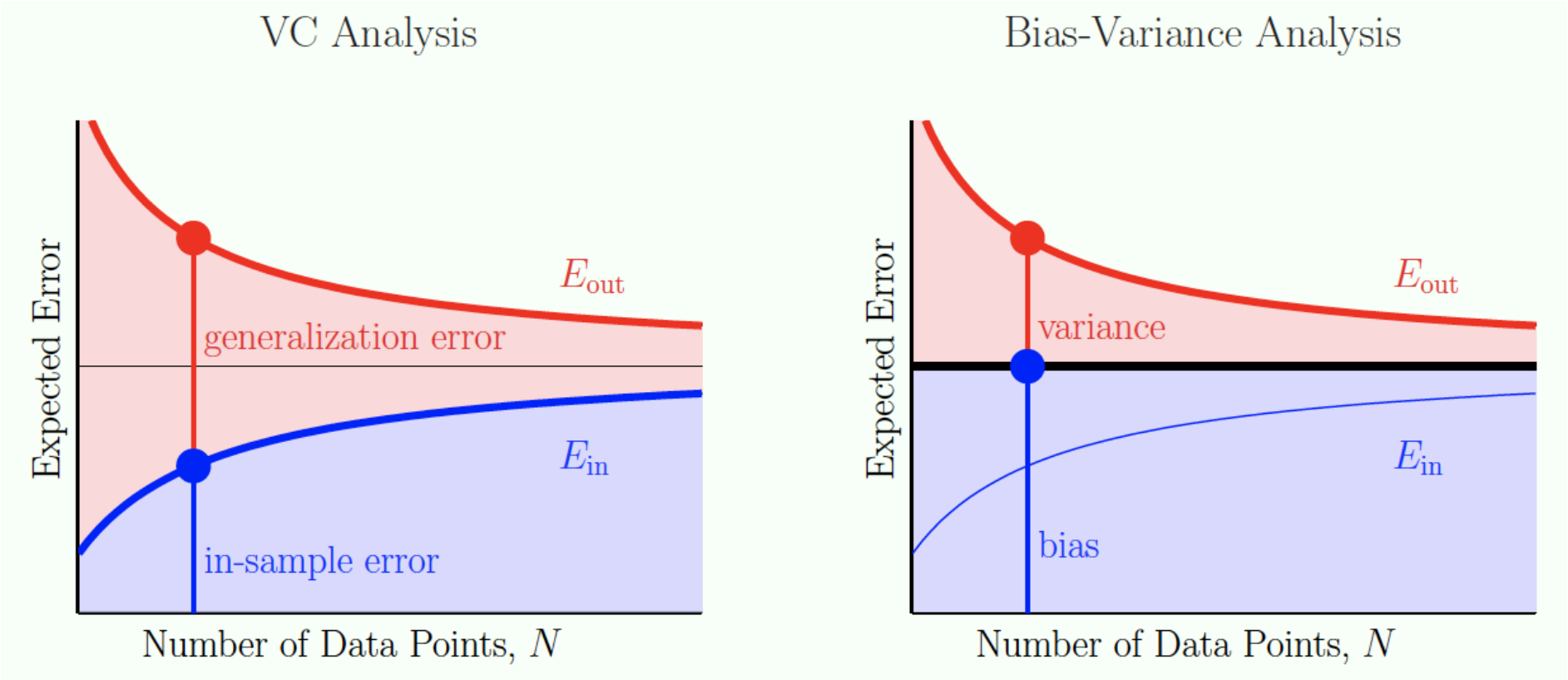
© Jiaming Mao

Learning Curve



© Jiaming Mao

Learning Curve



© Jiaming Mao

Noisy Targets

If y is not uniquely determined by x, i.e. if there does not exist a deterministic function f such that $y = f(x)$, then the relation between x and y needs to be described by a joint-distribution $p(x, y) = p(y|x)\, p(x)$.

Three approaches to learning and prediction when y is "noisy"[12]:

1. Learn $p(y|x)$: in this case we have a **target distribution** rather than a target function[13].
2. Find a deterministic function f such that $y = f(x) + e$, where e is an error term, and let f be our target function.
3. Let $p(x, y)$ be our target distribution, from which we can calculate $p(y|x) = \frac{p(x,y)}{p(x)}$.

[12] We say y is a noisy target when conditional on x, y is not completely determined.
[13] In Bayesian terms, $p(y|x)$ is the **posterior distribution** of y.

© Jiaming Mao

Learning p(y|x)

- To learn $p(y|x)$, let the hypothesis set $\mathcal{H}$ be a set of conditional probability distributions: $\mathcal{H} = \{q_1(y|x), q_2(y|x), \ldots\}$.
  - $\mathcal{H}$ is said to be a **probabilistic model**[14].
- Goal: select a $q(y|x) \in \mathcal{H}$ that approximates $p(y|x)$ well.
- What is a suitable loss function for quantifying how well $q(y|x)$ approximates $p(y|x)$?
  - Need: a measure of (dis)similarity between probability distributions.

[14] In general, hypothesis sets consisting of (conditional) probability distributions are called probabilistic models.

© Jiaming Mao

KL Divergence

Let p and q be two distributions of x. The **Kullback-Leibler (KL) divergence** of q from p[15], is defined as

$$D_{KL}(p||q) = \sum_x \log\left(\frac{p(x)}{q(x)}\right) p(x) \tag{8}$$

, or in the case of continuous random variables,

$$D_{KL}(p||q) = \int \log\left(\frac{p(x)}{q(x)}\right) p(x)\, dx \tag{9}$$

[15] Also called **relative entropy**. See Appendix II for an introduction to information theory, entropy, and KL divergence.

© Jiaming Mao

KL Divergence as Loss Function

KL divergence can be interpreted as a measure of dissimilarity between two distributions: $D_{KL}(p||q) \geq 0$ if and only if $p = q$[16].

$$\begin{aligned} D_{KL}(p||q) &= \mathbb{E}_{x\sim p(x)}\left[\log p(x) - \log q(x)\right] \\ &= -\mathbb{E}_{x\sim p(x)}\left[\log q(x) - \log p(x)\right] \\ &\geq -\log \mathbb{E}_{x\sim p(x)}\left[\frac{q(x)}{p(x)}\right] \\ &= -\log \int \frac{q(x)}{p(x)} p(x)\, dx = 0 \end{aligned}$$

Therefore, KL divergence can be used as a loss function to quantify the difference between probability distributions.

[16] Note that KL divergence is not symmetric: $D_{KL}(p||q) \neq D_{KL}(q||p)$. Therefore it is not a proper distance measure.

© Jiaming Mao

KL Divergence as Loss Function

Now suppose we are given data $\mathcal{D} = \{x_1, \ldots, x_N\} \sim^{i.i.d.} p(x)$ and want to learn $p(x)$ based on $\mathcal{D}$.

For any hypothesis distribution $q(x)$, using KL divergence as a loss function, we have:

$$E_{out}(q) = \mathbb{E}[\log p(x) - \log q(x)] \tag{10}$$

$$E_{in}(q) = \frac{1}{N} \sum_{i=1}^{N} (\log p(x_i) - \log q(x_i)) \tag{11}$$

© Jiaming Mao

KL Divergence as Loss Function

Since p is fixed, choosing a q to minimize (10) and (11) is equivalent to minimizing[17]:

$$E_{out}(q) = -\mathbb{E}[\log q(x)] \tag{12}$$

$$E_{in}(q) = -\frac{1}{N}\sum_{i=1}^{N} \log q(x_i) \tag{13}$$

- **cross-entropy loss**: $\ell(q(x), p(x)) = -\log q(x)$

[17] (12) and (13) are the out-of-sample and in-sample expressions for **cross entropy**. Given a fixed true distribution p, minimizing the KL divergence of any distribution q from p is the same as minimizing their cross entropy. See Appendix II.

© Jiaming Mao

Maximum Likelihood

Given observed data $\mathcal{D}$ and a probability distribution q, the **likelihood function** is defined as the probability of observing $\mathcal{D}$ according to q:

$$\mathcal{L}(q) = \Pr_q(\mathcal{D}) = \prod_{i=1}^{N} q(x_i) \tag{14}$$

$\Rightarrow$ the log likelihood function

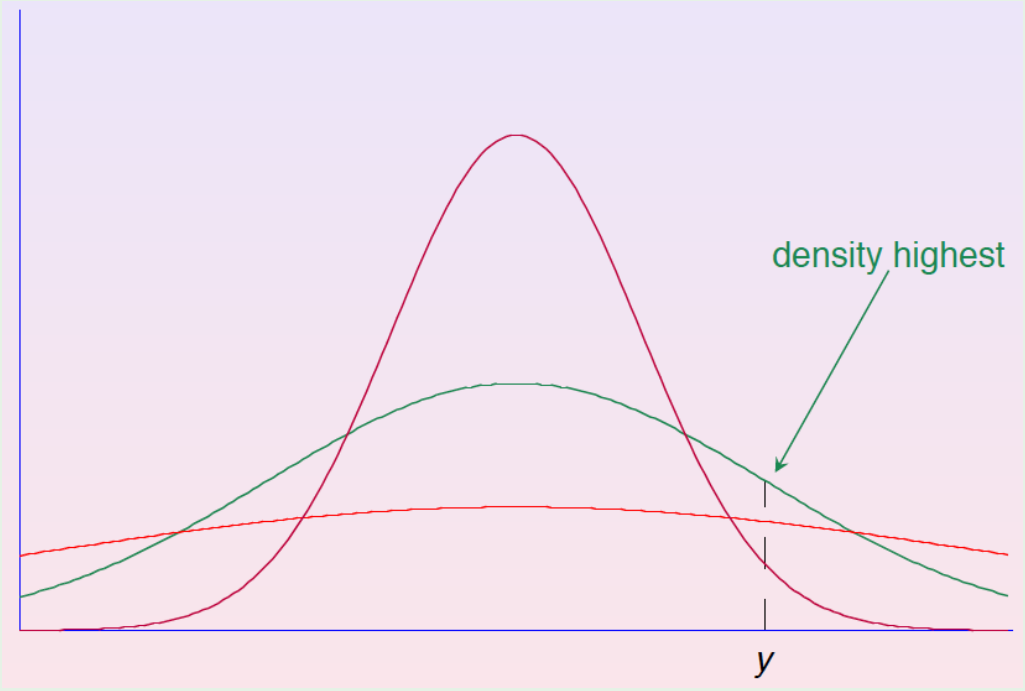
$$\log \mathcal{L}(q) = \sum_{i=1}^{N} \log q(x_i) \tag{15}$$

Let the hypothesis set $\mathcal{H}$ be a set of probability distributions. The **maximum likelihood estimation (MLE)** method chooses a distribution from $\mathcal{H}$ that maximizes the (log) likelihood function.

© Jiaming Mao

Maximum Likelihood

Suppose we only observe a single data point, y, drawn from an underlying distribution. We want to learn the underlying distribution based on this one data point. Our hypothesis set consists of the following three distributions:



Then according to MLE, we would choose the green distribution.

© Jiaming Mao

Maximum Likelihood as Minimum KL Divergence

Minimizing the empirical KL divergence (cross entropy) is equivalent to maximizing the (log) likelihood function.

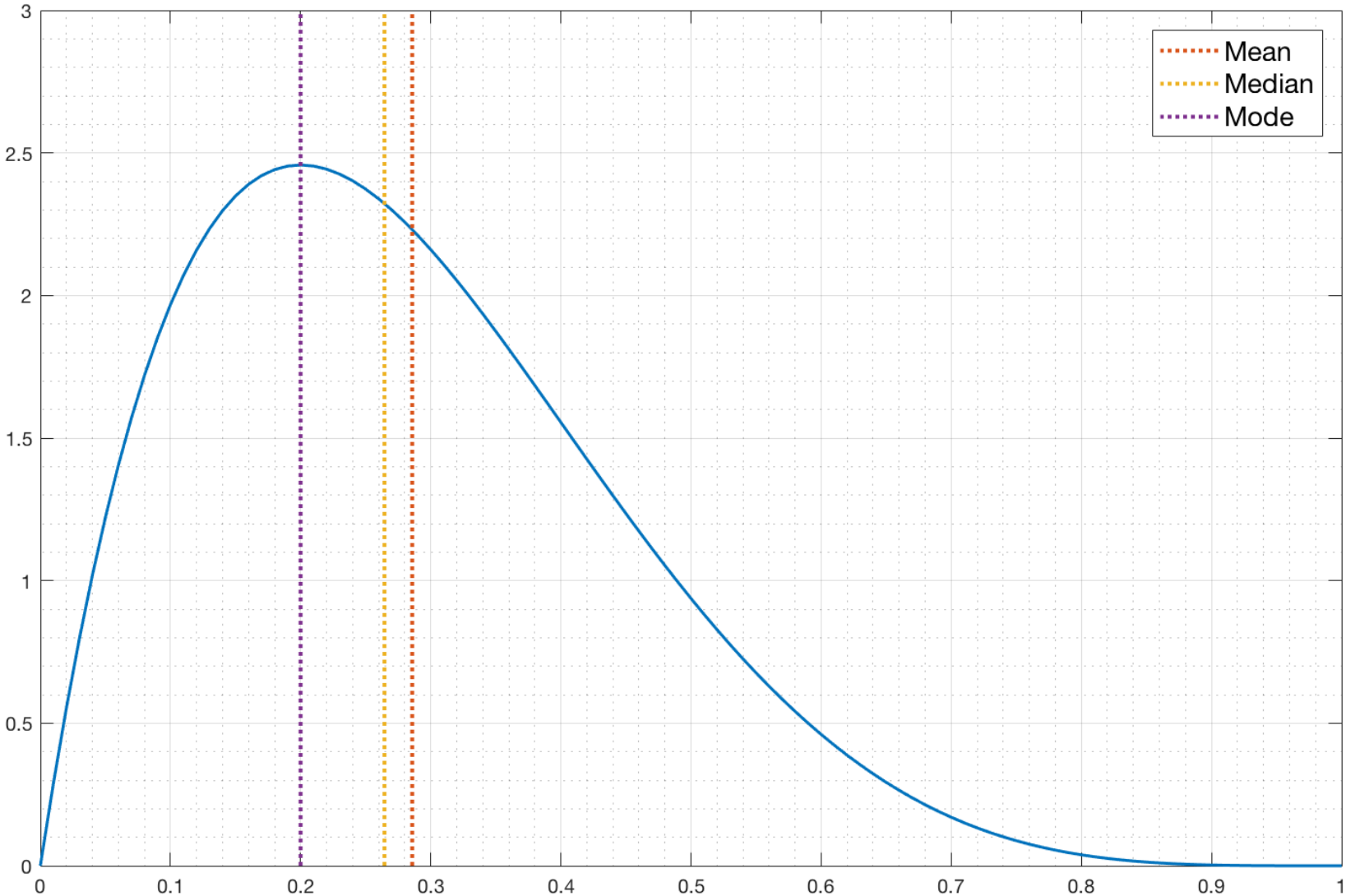
© Jiaming Mao

Learning p(y|x)

- Now suppose from a hypothesis set $\mathcal{H} = \{q_1(y|x), q_2(y|x), \ldots\}$, we have selected a $q(y|x)$ to approximate $p(y|x)$ by minimizing the KL divergence. Let's write $q(y|x)$ as $\widehat{p}(y|x)$.
- Armed with $\widehat{p}(y|x)$ – our estimate of $p(y|x)$ – how should we make a prediction of y given a value of x?
- For continuous y, let $\widehat{y}(x)$ denote our prediction of y given x. There are many choices: $\widehat{y}(x)$ can be
  - mean of $\widehat{p}(y|x)$
  - median of $\widehat{p}(y|x)$
  - mode of $\widehat{p}(y|x)$
  - ...
- It depends on the loss function that we use.

© Jiaming Mao

Learning p(y|x)



A hypothetical $\widehat{p}(y|x)$. What should $\widehat{y}(x)$ be?

© Jiaming Mao

Learning p(y|x)

Given $\widehat{p}(y|x)$, let $\widehat{y}(x)$ be the solution to

$$\widehat{y}(x) = \arg\min_{c} \mathbb{E}_{\widehat{p}(y|x)} [\ell(y, c)|\, x]$$

Then

$$\begin{aligned} \ell(y, c) = (y - c)^2 &\Rightarrow \widehat{y} = \mathbb{E}[y|x] \\ \ell(y, c) = |y - c| &\Rightarrow \widehat{y} = \text{Median}(y|x) \\ \ell(y, c) = \mathcal{I}(y \neq c) &\Rightarrow \widehat{y} = \text{Mode}(y|x) \end{aligned}$$

, where the mean, median, and mode are with respect to $\widehat{p}(y|x)$.

© Jiaming Mao

Learning p(y|x)

For discrete or categorical y, a common choice is to use the $0-1$ loss[18]:

		y			
		1	2	$\cdots$	K
$\widehat{y}$	1	0	1	$\cdots$	1
	2	1	0	$\cdots$	1
	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
	K	1	1	$\cdots$	0

$$\ell(y, \widehat{y}) = \mathcal{I}(y \neq \widehat{y}) \text{ for } y \in \{1, \ldots, K\}$$

[18] In the classification setting, the $0-1$ loss is also called the **misclassification loss**.

© Jiaming Mao

Learning p(y|x)

Given $\widehat{p}(y|x)$, using the $0-1$ loss for prediction, we have:

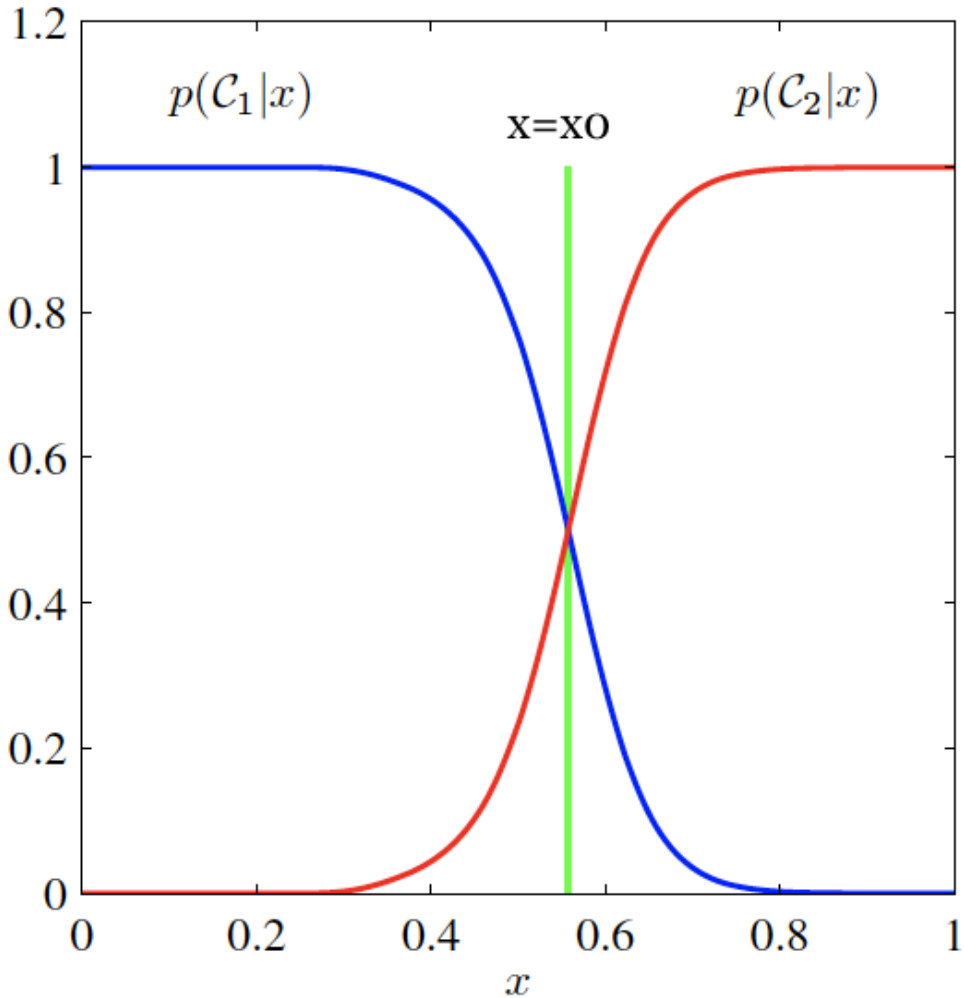
$$\begin{aligned}\widehat{y}(x) &= \underset{c\in\{1,\ldots,K\}}{\arg\min}\ \mathbb{E}_{\widehat{p}(y|x)}\left[\mathcal{I}(y\neq c)|\, x\right] \\ &= \underset{c\in\{1,\ldots,K\}}{\arg\min}\ \widehat{p}(y\neq c|x) \\ &= \underset{c\in\{1,\ldots,K\}}{\arg\max}\ \widehat{p}(y=c|x) \end{aligned} \tag{16}$$

, i.e., we predict y to be the value (class, category) that has the highest posterior probability[19]. This is called the **Bayes classifier**.

[19] according to the estimated $\widehat{p}(y|x)$.

© Jiaming Mao

Learning p(y|x)



$y \in \{C_1, C_2\}$. The Bayes classifier classifies y to be C_1 for $x < x_0$ and C_2 for $x > x_0$. The green line $x = x_0$ is called a **decision boundary**.

© Jiaming Mao

Learning p(y|x)

The loss function we use here is separate and can be different from the loss function that we use for learning $p(y|x)$. This is because for predicting noisy targets, we essentially have two stages:

1. Learning $p(y|x)$
2. Making prediction of y based on the estimated $p(y|x)$

These two stages are called learning and prediction[20].

[20] Also called inference and decision.

© Jiaming Mao

Decision Theory

How to make a prediction of y based on its probability distribution is a subject of decision theory, which is concerned with how to make optimal decisions given the appropriate probabilities.

Fingerprint Verification

Consider the problem of fingerprint verification. Let $y \in \{-1, 1\}$ denote whether the fingerprint belongs to the person of interest or not. Let $\widehat{y}$ be our prediction. There are two types of error we can make here:

		y	
		$+1$	-1
$\widehat{y}$	$+1$	no error	**false positive**
	-1	**false negative**	no error

© Jiaming Mao

Decision Theory

Fingerprint Verification

Loss functions can be used to control which type of error we want to minimize: the overall error rate, the false positive rate (FPR), or the false negative rate (FNR).

		y	
		$+1$	-1
$\widehat{y}$	$+1$	0	1
	-1	10	0

Supermarket $\ell(y, \widehat{y})$

		y	
		$+1$	-1
$\widehat{y}$	$+1$	0	1000
	-1	1	0

CIA $\ell(y, \widehat{y})$

The choice of $\ell(y, \widehat{y})$ depends on our needs.

© Jiaming Mao

Decision Theory

Fingerprint Verification

If $\ell(y, \widehat{y}) = \mathcal{I}(y \neq \widehat{y})$, then the decision rule is the Bayes classifier[a]

$$\text{predict } \widehat{y} = \begin{cases} 1 & \text{if } p(y=1) \geq p(y=0) \\ 0 & \text{if } p(y=1) < p(y=0) \end{cases}$$

, which minimizes the overall error rate.

[a] Assuming we know $p(y)$.

© Jiaming Mao

Learning f

- The second approach to learning and prediction when y is "noisy" is to find a deterministic function f such that $y = f(x) + e$, and let f be our target function (see page 49).
- Then let $\widehat{f}$ be our estimated f. Our prediction of y for any value of x will just be $\widehat{y} = \widehat{f}(x)$.
- What should f be? Ideally, f should be the function that best predicts y in the *underlying population*. Then we try to learn this f using our observed sample $\mathcal{D}$. Finally, we use $\widehat{f}$ to make predictions of y given x.
- This approach combines the two stages – learning and prediction – into one problem: directly learning a function f that maps each x into a prediction of y.

© Jiaming Mao

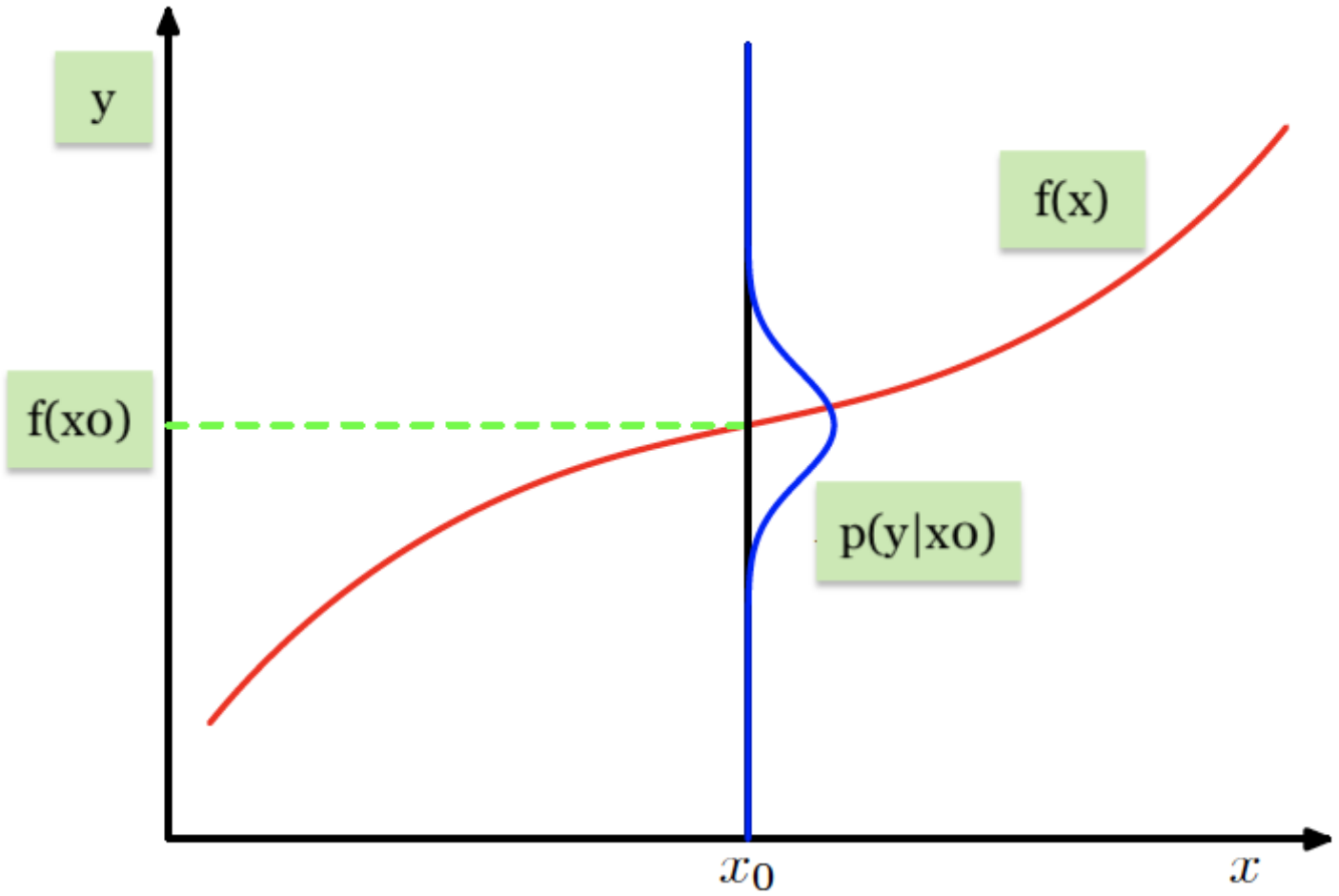
Learning f

- What is the function that produces the best prediction of y given x in the underlying population?
- The answer, again, depends on the loss function, i.e. on what we mean by "best."
- A common choice for continuous y is to use the squared-error loss, which $\Rightarrow f(x) = \mathbb{E}[y|x]$[21].
- The conditional expectation function $\mathbb{E}[y|x]$ is known as the **regression function**.

[21] Thus in this approach, instead of learning $p(y|x)$, we only learn a *moment* of $p(y|x)$, which is $\mathbb{E}[y|x]$.

© Jiaming Mao

Learning f

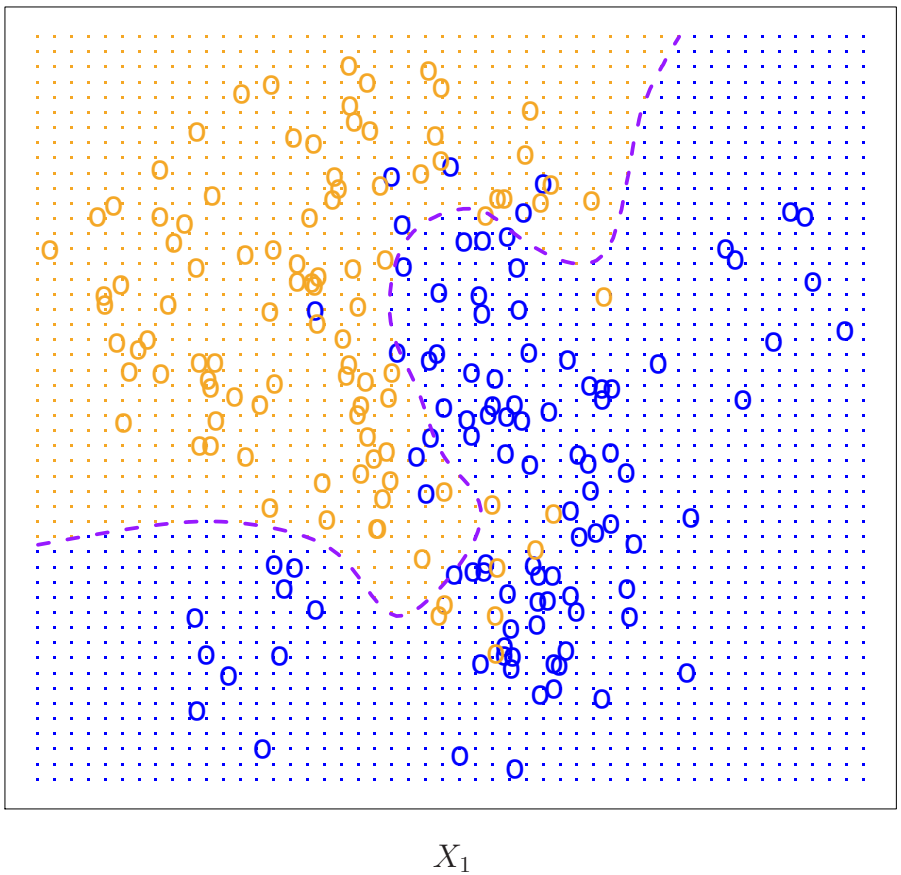


The regression function $f(x)$, which minimizes the expected squared error loss, is given by the mean of the conditional distribution $p(y|x)$.

© Jiaming Mao

Learning f

When y is discrete or categorical, this approach tries to learn the decision boundaries directly.



$x = (x_1, x_2)$, $y \in \{\text{orange, blue}\}$

Rather than estimating $p(y|x)$ and using it to derive a decision rule (e.g., the Bayes classifier), this approach focuses on learning directly the f (here the purple boundary) that best separates $y = \text{orange}$ and $y = \text{blue}$.

© Jiaming Mao

Learning p(x,y)

- The third approach is to learn the entire joint distribution $p(x, y)$ (see page 49).

- Let $\widehat{p}(x, y)$ be an estimate of $p(x, y)$. Once we have $\widehat{p}(x, y)$, we can use it to calculate $\widehat{p}(y|x)$, which in turn, would allow us to make a prediction of y given x.

© Jiaming Mao

Generative vs. Discriminative Models

- Models of the joint distribution $p(x, y)$ are called **generative models**[22], while models of $p(y|x)$ or $f(x)$ are called **discriminative models**[23].

- While discriminative models are mainly used for prediction tasks, generative models allows us to do more than just making predictions of y given x. We can, for example, generate new data points $\{(x_i, y_i)\}$ by drawing from $\widehat{p}(x, y)$. These new data points are called **synthetic data**, since they are not real, observed data. The process of generating synthetic data is called **simulation**.

[22] Approach 3 on page 49
[23] Approach 1 and 2 on page 49

© Jiaming Mao

Scientific Models

- **Scientific models**[24] are an important type of generative models that describe the **causal mechanisms** that generate $p(x, y)$.
- While scientific models can be used for prediction, the goal of learning causal mechanisms is distinct from the goal of prediction.

[24] Also called **causal models**.

© Jiaming Mao

Scientific Models

Scientific vs. Statistical Model

If you want to predict where Mars will be in the night sky[a], you may do very well with a model in which Mars revolves around the Earth. You can estimate, from data, how fast Mars goes around the Earth and where it should be tonight. But the estimated model does not describe the actual causal mechanisms. Nor does it need to: if our only goal is prediction, then we often do not need a scientific model.

[a] This example is taken from Shalizi (2019).

© Jiaming Mao

Scientific Models

- Because scientific models describe causal mechanisms, what we learn from one set of data $\mathcal{D} \sim p(x, y)$ can be potentially used to explain and predict data drawn from another distribution, say $p(u, v)$, if $\{x, y\}$ and $\{u, v\}$ share similar underlying causal mechanisms.
  - In other words, what we learn from one observed phenomenon can be used to explain and predict other related phenomena.
  - For example, we can learn individuals' risk aversion from their investment behavior, which in turn, can help explain and predict their career choices.

© Jiaming Mao

Scientific Models

- Good scientific models[25] can potentially deliver better predictive performance than statistical models trained on single data sets, because they can be learned from a combination of data from various sources that share the same underlying causal mechanisms.
  - Apples falling down trees and the earth orbiting around the sun both inform us of the gravitational constant.

[25] Think of quantum mechanics!

© Jiaming Mao

Appendix I: Growth Function

- Consider binary target functions and hypothesis sets that contain $h : \mathcal{X} \to \{-1, +1\}$[26].
- Let $\mathcal{H}(x_1, \ldots, x_N) = \{(h(x_1), \ldots, h(x_N)) | \, h \in \mathcal{H}\}$ denote the dichotomies generated by $\mathcal{H}$ on $x_1, \ldots, x_N \in \mathcal{X}$.

[26] The following analysis is all based on binary target functions.

© Jiaming Mao

Appendix I: Growth Function

Definition

The **growth function** for a hypothesis set $\mathcal{H}$ is defined by

$$m_{\mathcal{H}}(N) = \max_{x_1,\ldots,x_N \in \mathcal{X}} |\mathcal{H}(x_1, \ldots, x_N)|$$

, i.e., $m_{\mathcal{H}}(N)$ is the maximum possible number of dichotomies $\mathcal{H}$ can generate on a data set of N points[a].

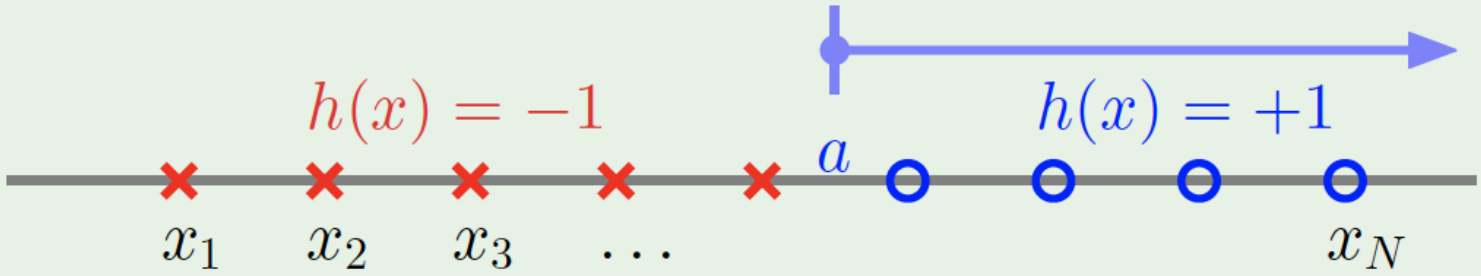
Note: $m_{\mathcal{H}}(N) \leq 2^N$. If $\mathcal{H}$ is capable of generating all possible dichotomies on $x_1, \ldots, x_N$, then $\mathcal{H}$ **shatters** $x_1, \ldots, x_N$, in which case $m_{\mathcal{H}}(N) = 2^N$.

[a] rather than over the entire input space $\mathcal{X}$.

© Jiaming Mao

Appendix I: Growth Function

Positive Rays



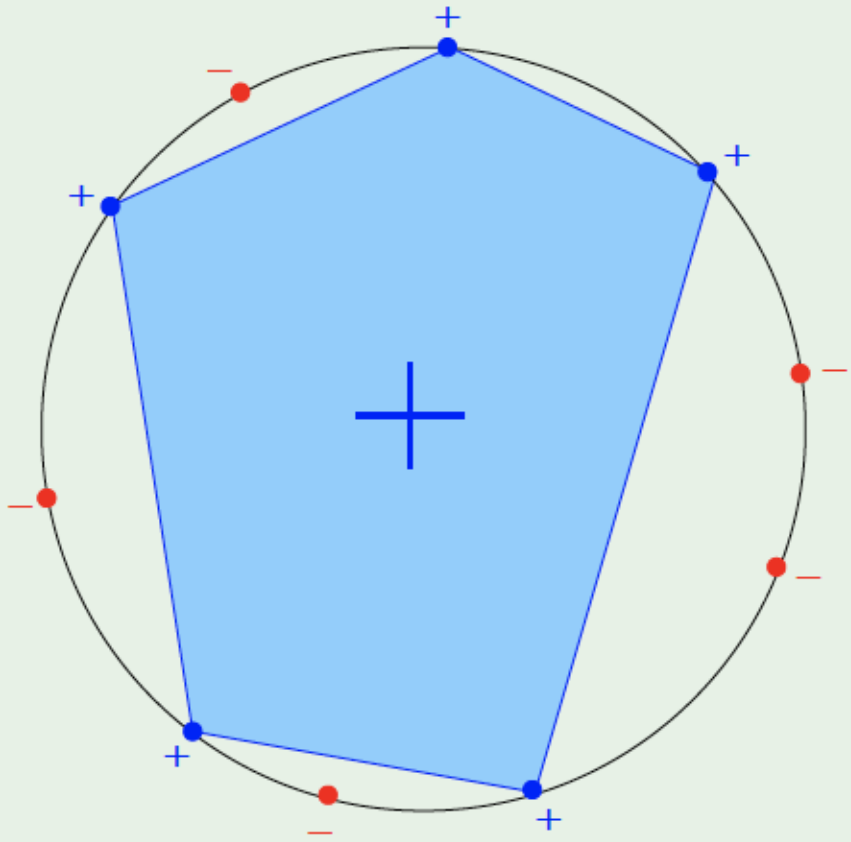
- $\mathcal{H} = \{h(x) = sign(x - a)\}$
- There are $N + 1$ dichotomies depending on where you put a.
- $m_{\mathcal{H}}(N) = N + 1$

© Jiaming Mao

Appendix I: Growth Function

Convex Sets

- $\mathcal{H}$ consists of all $h : \mathbb{R}^2 \to \{-1, +1\}$ that are are positive inside some convex set and negative elsewhere.
- If N points lie on a circle, then any dichotomy on these points can be generated by an h that is positive inside the polygon that connects the $+1$ points. Hence the N points are *shattered* by $\mathcal{H}$.
- $m_{\mathcal{H}}(N) = 2^N$



© Jiaming Mao

Appendix I: VC Dimension

Definition

The **Vapnik-Chervonenkis (VC) dimension** of $\mathcal{H}$, denoted $d_{vc}(\mathcal{H})$, is the size of the largest data set that $\mathcal{H}$ can shatter.

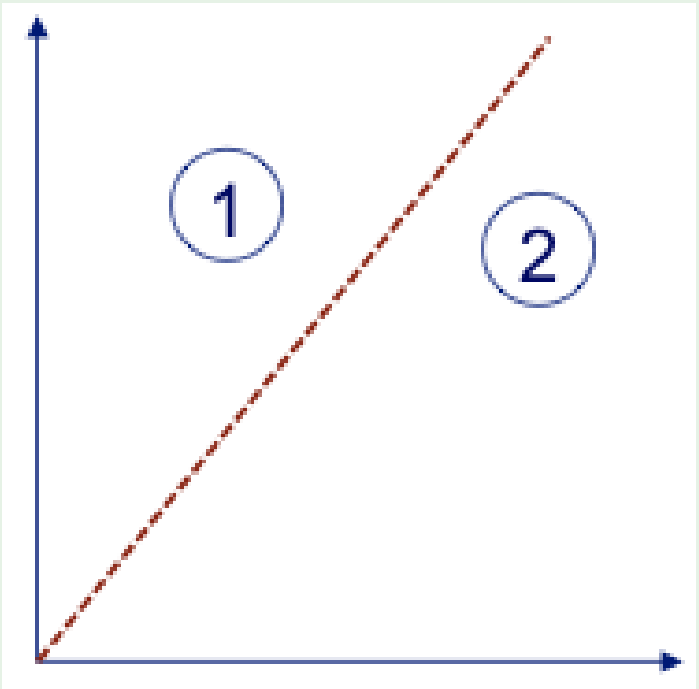
- $d_{vc}(\mathcal{H})$ is the largest value of N for which $m_{\mathcal{H}}(N) = 2^N$.
- If arbitrarily large finite sets can be shattered by $\mathcal{H}$, then $d_{vc}(\mathcal{H}) = \infty$.
- $\exists$ *some* shattered set of size $d \Rightarrow d_{vc}(\mathcal{H}) \geq d$.
- No set of size $d+1$ is shattered $\Rightarrow d_{vc}(\mathcal{H}) \leq d$.

© Jiaming Mao

Appendix I: VC Dimension

Hyperplanes in $\mathbb{R}^2$

- $\mathcal{H}$ is set of lines (linear separators) in $\mathbb{R}^2$
- can find can an h consistent with 2 data points no matter how they are labeled:

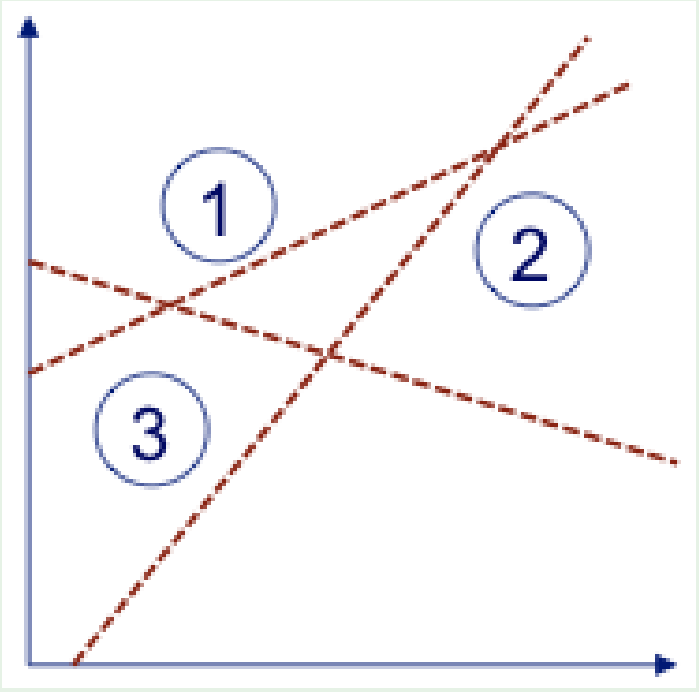


© Jiaming Mao

Appendix I: VC Dimension

Hyperplanes in $\mathbb{R}^2$

- can find can an h consistent with 3 non-collinear data points no matter labeling:

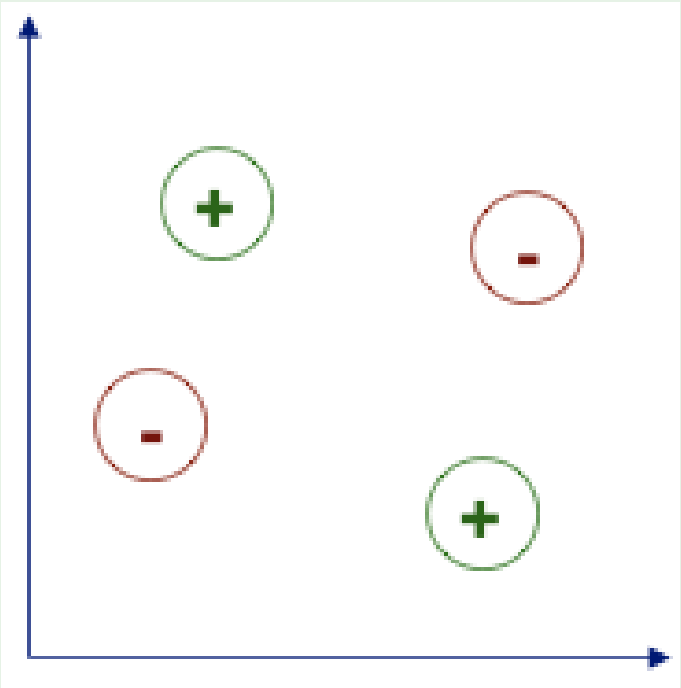


© Jiaming Mao

Appendix I: VC Dimension

Hyperplanes in $\mathbb{R}^2$

- *cannot* find can an h consistent with 4 data points for some labeling:



- Hence $d_{vc}(\mathcal{H}) = 3$[a].

[a] In general, $d_{vc}\left(\text{hyperplanes in } \mathbb{R}^d\right) = d + 1$

© Jiaming Mao

Appendix I: Sauer's Lemma

Sauer's lemma

If $d_{vc}(\mathcal{H}) < \infty$, then

$$m_{\mathcal{H}}(N) \leq \sum_{i=0}^{d_{vc}(\mathcal{H})} \binom{N}{i} \tag{17}$$

- If the VC dimension is finite, then $m_{\mathcal{H}}(N)$ can be bounded by a polynomial in N and the order of the polynomial is $d_{vc}(\mathcal{H})$.

© Jiaming Mao

Appendix I: Sauer's Lemma

We can prove that[27]

$$\sum_{i=0}^{d} \begin{pmatrix} N \\ i \end{pmatrix} \leq \begin{cases} N^d + 1 \\ \left(\frac{eN}{d}\right)^d \end{cases}$$

Therefore, $m_{\mathcal{H}}(N)$ can be further bounded by:

$$m_{\mathcal{H}}(N) \leq N^{d_{vc}(\mathcal{H})} + 1 \tag{18}$$

, or

$$m_{\mathcal{H}}(N) \leq \left(\frac{eN}{d_{vc}(\mathcal{H})}\right)^{d_{vc}(\mathcal{H})} \tag{19}$$

[27] The second inequality requires $N \geq d$.

© Jiaming Mao

Appendix II: Information Theory

Consider a random variable x. How much information is received when we observe a specific value of x?

- Depends on 'degree of surprise': a highly improbable value conveys more information than a very likely one.
- If we know an event is certain to happen, we would receive no information when we observe it happens.

© Jiaming Mao

Appendix II: Information Theory

Let $h(.)$ denote the information content of an event. $h(.)$ should satisfy

1. $h(a)$ should be inversely correlated with $p(a)$.
2. For two unrelated events a and b, such that $p(ab) = p(a)\,p(b)$, we should have $h(ab) = h(a) + h(b)$.

$\Rightarrow$ we can let:

$$h(a) = \log \frac{1}{p(a)}$$

© Jiaming Mao

Appendix II: Entropy

For a discrete random variable x with probability distribution $p(x)$, the average amount of information transmitted by x is:

$$\mathbb{H}(p) = E_p[h(x)] = \sum_x p(x) \log \frac{1}{p(x)}$$

$\mathbb{H}(p)$ is called the **entropy**[28] of probability distribution p .

- Distributions that are sharply peaked around a few values will have a relatively low entropy, while those that are spread more evenly across many values will have higher entropy.

[28] More precisely, **information entropy**, or **Shannon entropy**.

© Jiaming Mao

Appendix II: Entropy

- Historically, information entropy is developed to describe the average amount of information needed to specify the state of a random variable.
- Specifically, if we use base-2 logarithm in the definition of $\mathbb{H}(p)$, then $\mathbb{H}(p)$ is a lower bound on the average number of bits needed to encode a random variable with probability distribution p.
- Achieving this bound would require using an optimal coding scheme designed for p, which assigns shorter codes to higher probability events and longer codes to less probable events.

© Jiaming Mao

Appendix II: Entropy

- Suppose a random variable has 8 states, each being equally likely. Then we can code these 8 states as 000, 001, 010, 011, 100, 101, 110, 111. In this case, the average length of the code needed to encode the variable is 3, which is equivalent to its entropy $\mathbb{H} = 8 \times \frac{1}{8} \log_2 8 = 3$.
- If the probabilities of the 8 states are given by $\left(\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \frac{1}{64}, \frac{1}{64}, \frac{1}{64}, \frac{1}{64}\right)$, then the optimal coding scheme is 0, 10, 110, 1110, 111100, 111101, 111110, 111111. Under this coding scheme, the average length of the code needed to encode the variable is $\frac{1}{2} \times 1 + \frac{1}{4} \times 2 + \frac{1}{8} \times 3 + \cdots = 2$, which is equivalent to its entropy $\mathbb{H} = \frac{1}{2} \log_2 2 + \frac{1}{4} \log_2 4 + \frac{1}{8} \log_2 8 + \cdots = 2$.

© Jiaming Mao

Appendix II: Cross Entropy

If p is the distribution of x, but we use distribution q to describe x instead, then the average amount of information needed to specify x as a result of using q instead of p is

$$\mathbb{H}(p, q) = \mathbb{E}_p \left[\log \frac{1}{q(x)} \right] = \sum_x p(x) \log \frac{1}{q(x)}$$

$\mathbb{H}(p, q)$ is called the **cross entropy** of p and q. In information theory[29], it can be interpreted as the average number of bits needed to encode a random variable using a coding scheme designed for probability distribution q rather than the true distribution p.

[29] Using base-2 logarithm.

© Jiaming Mao

Appendix II: KL Divergence

The **relative entropy** of q with respect to p, or the **Kullback-Leibler (KL) divergence** of q from p, is defined as

$$\begin{aligned} D_{KL}(p||q) &= \mathbb{H}(p,q) - \mathbb{H}(p) \\ &= \sum_x \log\left(\frac{p(x)}{q(x)}\right) p(x) \end{aligned}$$

, or in the case of continuous random variables,

$$D_{KL}(p||q) = \int \log\left(\frac{p(x)}{q(x)}\right) p(x)\, dx$$

KL divergence represents the average *additional* information required to specify x as a result of using q instead of the true distribution p.

© Jiaming Mao

Appendix II: Coin Guess

As an example to illustrate the concepts of entropy, cross entropy, and relative entropy (KL divergence), let's play the following games[30]:

Game 1

I will draw a coin from a bag of 4 coins: a blue, a red, a green, and an orange coin. Your goal is to guess which color it is with the fewest questions.

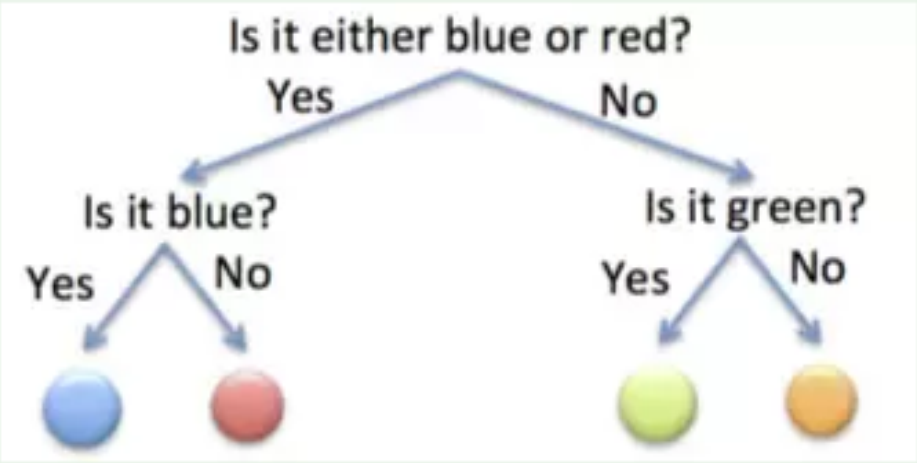
[30] Source of this example.

© Jiaming Mao

Appendix II: Coin Guess

Game 1

One of the best strategies is this:



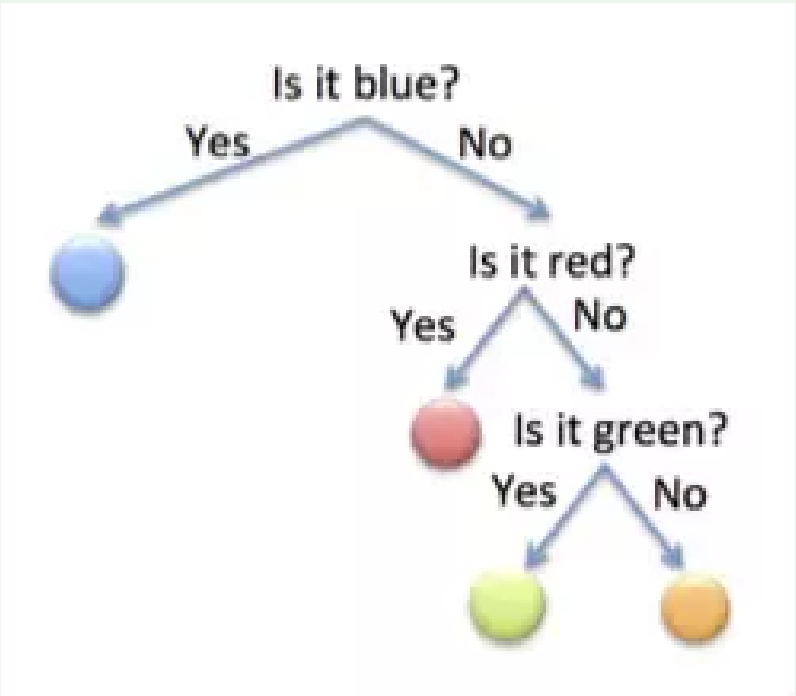
Using this strategy, the expected number of questions needed to guess the coin is 2. This is the entropy[29] of the probability distribution $p_1 = \left(\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}\right)$.

© Jiaming Mao

Appendix II: Coin Guess

Game 2

Now suppose the coins in the bag have the following distribution: 1/2 of them are blue, 1/4 are red, 1/8 are green, and 1/8 are orange. The optimal strategy now looks like this:



Under this strategy, the expected number of questions to guess a coin is $\frac{1}{2} \times 1 + \frac{1}{4} \times 2 + 2 \times \frac{1}{8} \times 3 = 1.75$. This is the entropy[29] of the probability distribution $p_2 = \left(\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{8}\right)$.

© Jiaming Mao

Appendix II: Coin Guess

Using Game 1 Strategy on Game 2

What if we still use the strategy for game 1 to play game 2?

Then the expected number of questions needed to guess the coin is $\frac{1}{2} \times 2 + \frac{1}{4} \times 2 + 2 \times \frac{1}{8} \times 2 = 2$. This is the cross entropy for using game 1 strategy (optimized for p_1) on game 2 (with probability distribution p_2).

Obviously, using Game 1 strategy on Game 2 is not optimal. The *additional* expected number of questions we need to ask as a result of not using the optimal strategy is $2 - 1.75 = 0.25$. This is the KL divergence of p_1 from p_2.

© Jiaming Mao

Appendix II: KL Divergence as Loss Function

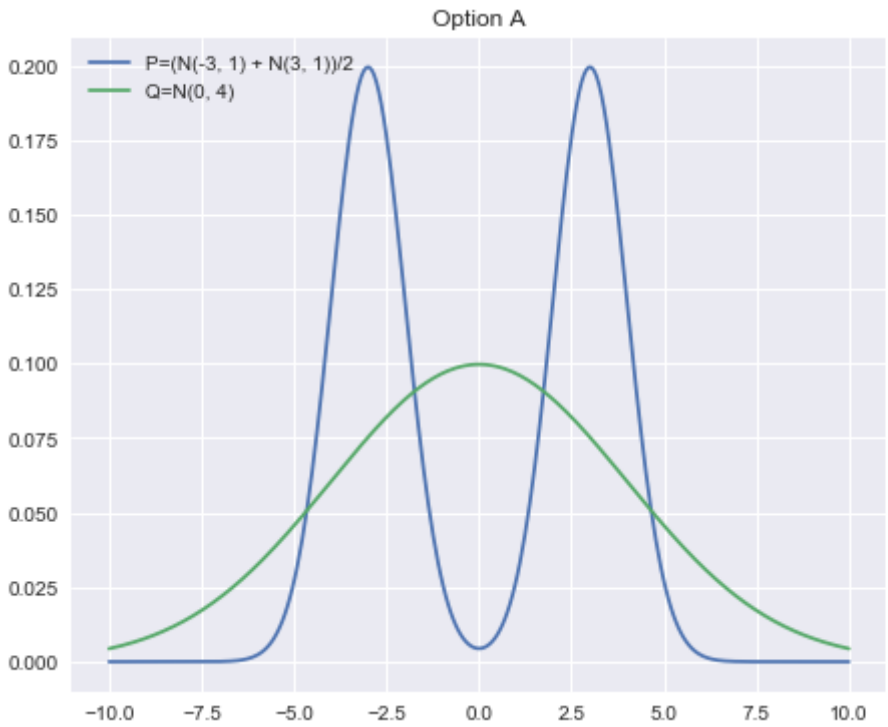
KL divergence can be interpreted as a measure of dissimilarity between two distributions. It satisfies $D_{KL}(p||q) \geq 0$ if and only if $p = q$.

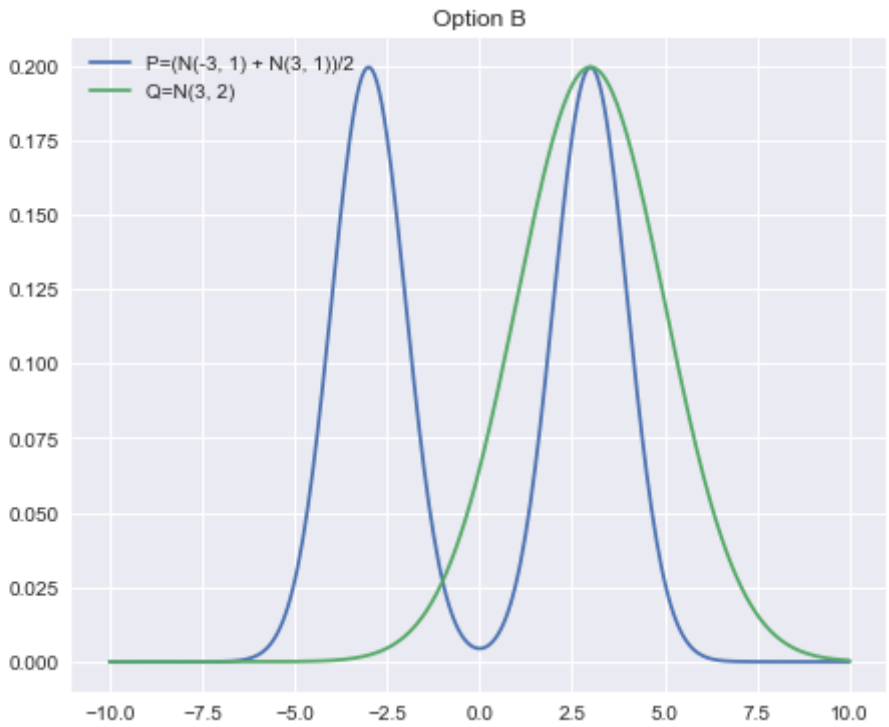
Therefore, KL divergence can be used as a loss function to quantify the difference between probability distributions.

© Jiaming Mao

Appendix II: KL Divergence as Loss Function

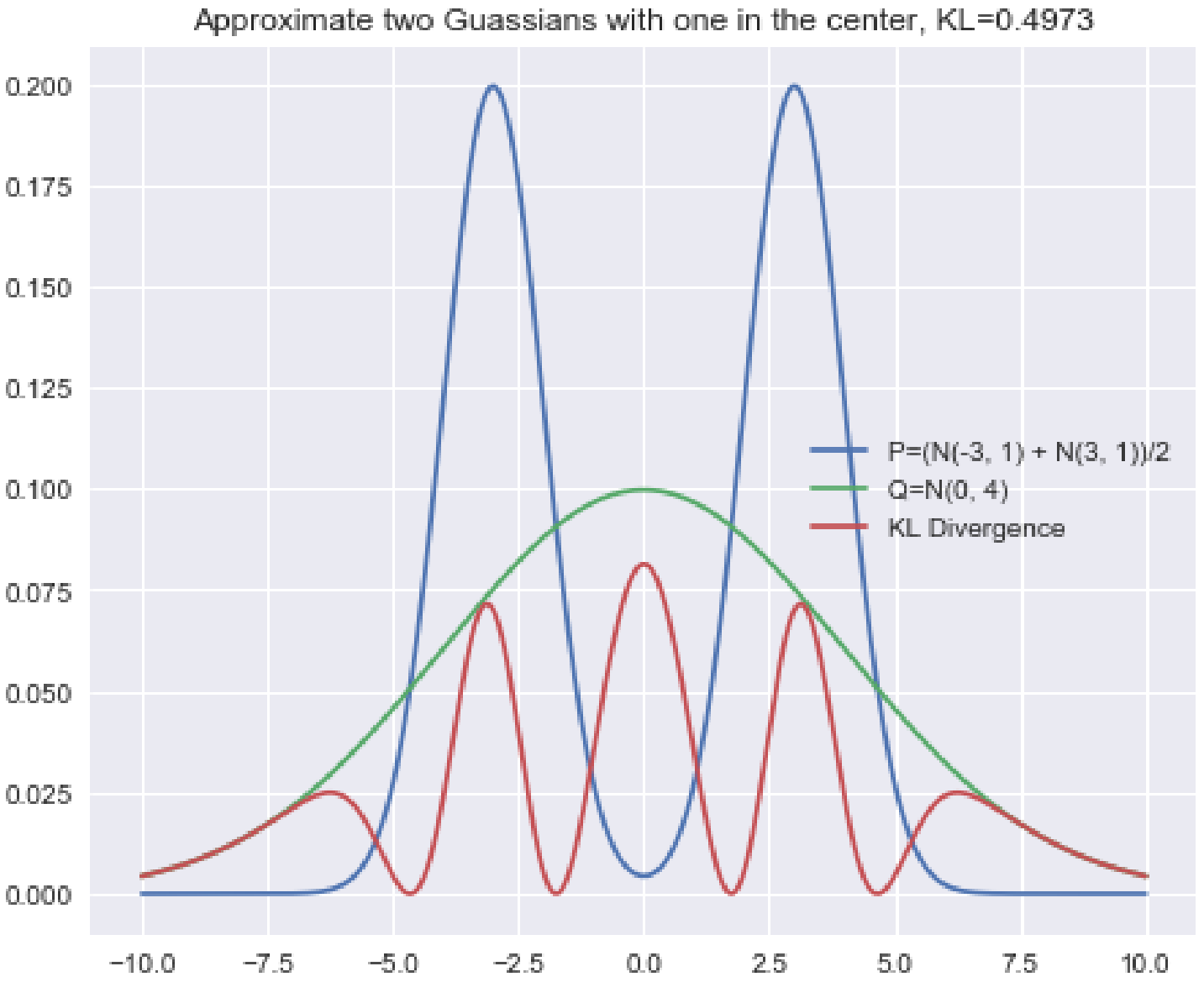
Which of the following Q distributions better approximates P?





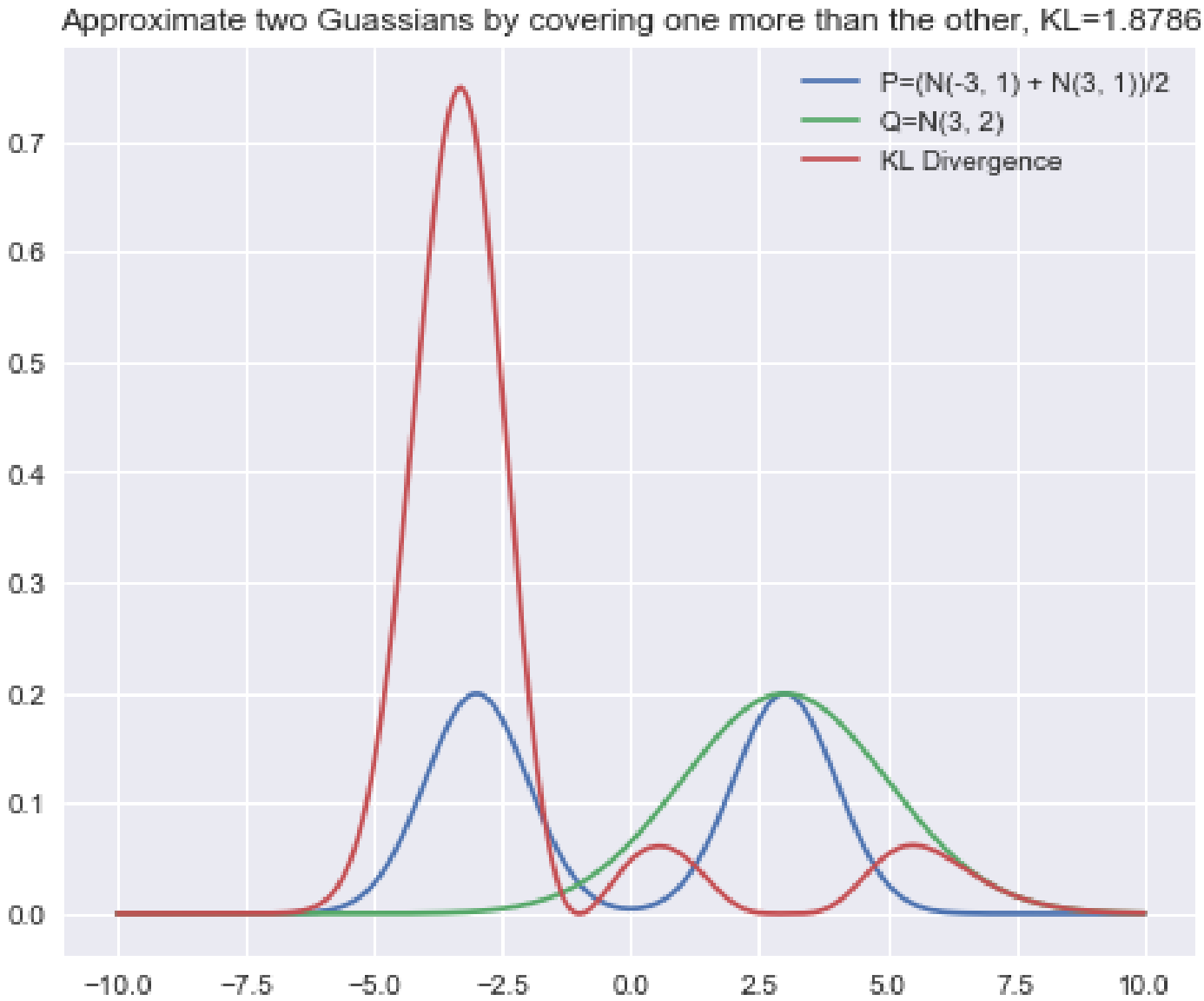
© Jiaming Mao

Appendix II: KL Divergence as Loss Function



© Jiaming Mao

Appendix II: KL Divergence as Loss Function



© Jiaming Mao

Acknowledgement

Part of this lecture is based on the following sources:

- Abu-Mostafa, Y. S., M. Magdon-Ismail, and H. Lin. 2012. *Learning from Data*. AMLBook.
- Bishop, C. M. 2011. *Pattern Recognition and Machine Learning*. Springer.
- James, G., D. Witten, T. Hastie, and R. Tibshirani. 2013. *An Introduction to Statistical Learning: with Applications in R*. Springer.
- Page, D. *Machine Learning*. Lecture at the University of Wisconsin-Madison, retrieved on 2018.01.01. [link]
- Shalizi, C. R. 2019. *Advanced Data Analysis from an Elementary Point of View*. Manuscript.

© Jiaming Mao